# Win32 Fortran Compiler Comparisons

Which is the best Fortran compiler? We're often asked that question, but there is no single answer. All compilers have strong and weak points, and your choice should depend on finding the best match between your requirements and a compiler's strengths. To help our customers make an informed decision, we have compiled a series of charts comparing the main contenders. The charts cover the questions we are most frequently asked, but avoid subjective issues, such as stability, quality of support and documentation etc. Please contact us if you need further advice in coming to your decision.


- Compiler Version Numbers
- Language - What extensions to standard Fortran do the compilers support?
- Diagnostic Capabilities - Can they spot programming errors?
- Third Party Support - What libraries and other packages can you use?
- Windows Programming - How do the compilers help you to create Windows programs.
- Fortran 77 Execution Time Benchmarks - 14 Fortran 77 benchmarks.
- Fortran 90 Execution Time Benchmarks - 10 Fortran 90 benchmarks.
- Processor Comparison - AMD Athlon vs Intel PIII.
- Debugger and IDE - What can you do with them?
- Special Features - Unique features of each compiler.
- Other Stuff - Links to C, Compilation benchmarks, O/S Support, Documentation etc.


We'd like to acknowledge the help of many people who have reviewed these tables at various stages in their construction, and made innumerable helpful comments, suggestions and corrections. Any remaining mistakes are, of course, entirely our responsibility; if you see any, please let us know, and we'll do our best to correct them. Our thanks to:-

- Bob Runyan, Lahey
- Craig Burley
- David Vallance, Salford Software
- Ken Hamilton
- Lawson Wakefield, ISS
- Mark Stevens, Salford Software
- Michael Ross, Intel
- Mike Delves, NA Software
- Peter Jacobson, Absoft
- Steve Lionel, Compaq
- Tim Prince
- Toon Moene
- Wood Lotz, Absoft

# Win32 Fortran Compiler Comparisons

## Compiler Version Numbers

Versions indicated below are the compiler versions used for testing - they are not necessarily the current version.

| *Code* | *Product* | *Version* |
|---|---|---|
| APF | Absoft Pro Fortran | 7.0 |
| CVF | Compaq Visual Fortran | 6.5 |
| FTN77 | Salford FTN77 | 4.02 |
| FTN95 | Salford FTN95 | 2.52 |
| G77 | G77 | 0.5.25 |
| INTEL | Intel Fortran Compiler | 5.0 |
| LF90 | Lahey LF90 | 4.50i |
| LF95 | Lahey/Fujitsu LF95 | 5.60e |
| NAS | N.A.Software FortranPlus | 2.2 |
| WAT | Watcom | 10.5 |

---

**(C) Polyhedron Software Ltd. (1998-2001)**

*Last Updated on 04/04/01*

*By John Appleyard*
*Email: john@polyhedron.com*

# Contact us for more information.

Please remember to state the name of the product that you are interested in.

---

Contact us by email, letter, fax or telephone.

Polyhedron Software Ltd.
Linden House
93 High Street
Standlake
WITNEY
OX29 7RH
United Kingdom

Tel +44 (0)1865-300579
Fax +44 (0)1865-300232
email wwwsales@polyhedron.com

For technical support on any of our products please email: support@polyhedron.com

**Please note that our post code has been changed.**

# Win32 Fortran Compiler Comparisons

## Supported Language

| Source | | ***Language Feature*** | ***APF77*** | ***APF95*** | ***CVF*** | ***FTN77*** | ***FTN95*** | ***G77*** | ***INTEL*** | ***LF90*** | ***LF95*** | ***NAS*** | ***WAT*** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | *Fortran 90* | No | Yes | Yes | No | Yes | No | Yes | Yes | Yes | Yes | No |
| | | *Fortran 95* | No | Yes | Yes | No | Yes | No | Yes | No | Yes | Yes | No |
| LANG1 | F90 | Lower case characters | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| LANG1 | F90 | In-line comment after ! | Yes | Yes | Yes | Yes[1] | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| LANG1 | F90 | Symbolic names with _ | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| LANG1 | F90 | "..." character constants | Yes | Yes | Yes | No | Yes | Yes | Yes | Yes | Yes | Yes | No |
| LANG1 | F90 | IMPLICIT NONE | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| LANG1 | F90 | NAMELIST | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| LANG1 | F90 | DO [label] .. [WHILE(..)] .. ENDDO | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| LANG1 | F90 | INCLUDE | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| LANG1 | F90 | 31 character variable names | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| LANG1 | | Tabbed source form | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| LANG1 | | Symbolic names with $ | Yes | Yes | Yes | Yes | Yes | Option | Yes | Yes | Yes | No | Yes |
| LANG1 | | Hollerith data | Yes | Yes | Yes | Yes | Yes | Yes | Yes | No | Yes | No | Yes |
| LANG1 | | DOUBLE COMPLEX | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| LANG1 | | Varying length for named COMMON | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| LANG1 | | Mix numeric and character in COMMON and EQUIVALENCE | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| Source | | ***Language Feature*** | ***APF77*** | ***APF95*** | ***CVF*** | ***FTN77*** | ***FTN95*** | ***G77*** | ***INTEL*** | ***LF90*** | ***LF95*** | ***NAS*** | ***WAT*** |
| LANG3 | | INTEGER*n | 1,2,4,8 | 1,2,4,8 | 1,2,4 | 1,2,4 | 1,2,4,8 | 1,2,4,8 | 1,2,4,8 | 1,2,4 | 1,2,4,8 | 1,2,4,8 | 1,2,4 |
| LANG3 | | LOGICAL*n | 1,2,4,8 | 1,2,4,8 | 1,2,4 | 1,2,4 | 1,2,4 | 1,2,4,8 | 1,2,4,8 | 1,4 | 1,2,4,8 | 1,2,4,8 | 1,4 |
| LANG3 | | REAL*n | 4,8 | 4,8 | 4,8 | 4,8 | 4,8,10 | 4,8 | 4,8,16 | 4,8 | 4,8,16 | 4,8,10 | 4,8 |
| | | | | | | | | | | | | | |
| LANG2 | | VAX style debug (D) lines | Yes | No | Yes | No | No | Yes[2] | Yes | No | Yes | No | Yes |
| LANG2 | | Binary Constants | B'1111' | B'1111' | B'1111' | B'1111' | B'1111' | B'1111' | B'1111' | B'1111'[4] | B'1111' | B'1111' | No |
| LANG2 | | Octal Constants | O'77' | O'77' | O'77' | O'77' | O'77' | O'77' | O'77' | O'77'[4] | O'77' | O'77' | '77'O |
| LANG2 | | Hex Constants | Z'FF' | Z'FF' | Z'FF' | Z'FF' | Z'FF' | Z'FF' | Z'FF' | Z'FF'[4] | Z'FF' | Z'FF' | 'FF'X |
| LANG2 | | C style string constants | Option | '\n'C | '\n'C | No | No | Option | Yes | No | No | No | ? |
| LANG2 | | VAX style STRUCTURE, RECORD, UNION etc. | Yes | Yes | Yes | No | No | No | Yes | No | Yes | No | Yes |
| LANG2 | | Initialization in TYPE statements | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| LANG2 | F90 | Intrinsics in PARAMETER constants | No | Yes | Yes | No | Yes | No | Yes | Yes | Yes | Yes | No |
| LANG2 | | ENCODE and DECODE | Yes | Yes | Yes | Yes | No | No | Yes | No | Yes | No | No |
| LANG2 | F90 | List-directed I/O on internal files | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| LANG2 | | variable format expressions using <> | No | No | Yes | No | No | Yes[4] | Yes[9] | No | Yes | No | No |
| LANG2 | | \ edit descriptor | Yes | No | Yes | Yes | Yes | No | Option | Yes | Yes | No | Yes |
| LANG2 | | Q edit descriptor | Yes | No | Yes | No | No | No | Yes | Yes | Yes | No | No |
| LANG2 | | $ edit descriptor | Yes | Yes | Yes | Yes | No | Yes | Yes | Yes | Yes | No | Yes |
| | | in line assembler | No | No | No | Yes | Yes | No | No | No | No | No | Yes |
| LANG2 | | CRAY Pointers | Yes | Yes | Yes | No | No | No | Yes | No | Yes | No | No |
| LANG2 | F90 | Bit Manipulation intrinsics | Yes | Yes | Yes | Yes[5] | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | F90 | Automatic Arrays | No | Yes | Yes | No | Yes | Yes | Yes | Yes | Yes | Yes | No |
| | | OPEN for "Transparent" I/O (e.g. FORM='BINARY') | Yes | Yes | Yes | Yes | Yes | No | Yes | Yes | Yes | No | ? |
| Source | | ***Language Feature*** | ***APF77*** | ***APF95*** | ***CVF*** | ***FTN77*** | ***FTN95*** | ***G77*** | ***INTEL*** | ***LF90*** | ***LF95*** | ***NAS*** | ***WAT*** |
| | | Get Command Line | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| | | Get Environment Variable | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| | | Invoke External command | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes[8] | Yes |
| | F90 | Date, Time, CPU Time | Yes | Yes[6] | Yes[6] | Yes | Yes[6] | Yes | Yes[6] | Yes[6] | Yes | Yes | Yes |
| | | Get Files in Directory | Yes[7] | Yes[7] | Yes | Yes | Yes | No | Yes | Yes[7] | Yes[7] | Yes[7] | Yes[7] |
| | | Get File Size, Date, Attributes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes[7] | Yes[7] | Yes[7] | Yes[7] |
| | | VAX style system intrinsics (SECNDS etc.) | Yes | Yes | Yes | No | No | Yes | Yes | No | Yes | No | No |
| | | Unix style system library (getenv, etime etc.) | Yes | Yes | Yes | No | No | Yes | Yes | No | Yes | Yes | ? |
| | | Posix style library (pxfputc, pxfopen etc.) | Yes[10] | Yes[10] | No | No | No | No | Yes | No | No | No | No |

| ***Notes*** | |
|---|---|
| 1 | FTN77 does not support end-of-line comments starting before column 7 |
| 2 | G77 always treats D-lines as comments |
| 3 | LF90 Binary, Octal and Hex constants may only be used in DATA statements |
| 4 | But only with constants and PARAMETERs |
| 5 | The Salford F77 compiler does not use MIL-STD-1753 naming conventions for bit manipulation routines |
| 6 | Fortran 90 intrinsics can be used. |
| 7 | The Win32 API call must be used |
| 8 | Using the NAS_SYSTEM module |
| 9 | You cannot use variable expressions in formats generated at runtime |
| 10 | Via Microsoft Interix. Windows 2000 or Windows NT must also be installed |

**(C) Polyhedron Software Ltd. (1998-2001)**

*Last Updated on 15/03/01*
*By John Appleyard*
*Email: john@polyhedron.com*

# Win32 Fortran Compiler Comparisons

## Diagnostic Capabilities

Download Diagnostic test source code (diagnose.zip 14K)

| Source | Run-time Error | APF77 | APF95 | CVF | FTN77 | FTN95 | G77 | INTEL | LF90 | LF95 | NAS | WAT |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Percentage Passes[1] | 19% | 33% | 54% | 91% | 96% | 40% | 60% | 42% | 90% | 48% | 34% |
| | Benchmark TFFT.for execution time with diagnostic switches (seconds)[2] | 47.14 | 50.42 | 70.40 | 93.55 | 124.11 | 86.79 | 79.48 | 66.14 | 2441.99 | 72.90 | 294.81 |
| ARG1 | argument mismatch - same file | No | No | Yes[3] | Yes[5] | Yes[5] | Yes[3] | Yes[5] | Yes[3] | Yes[4] | Yes[4] | No |
| ARG2 | argument mismatch - different file[6] | No | No | No[7] | Yes | Yes | No | Yes[5] | No[7] | Yes | Yes[4] | No |
| ARG3 | Illegal assignment to constant argument - no INTENT specified | No | No | No | Yes | Yes | No | No | No | Yes | No | No |
| ARG4 | Illegal assignment to constant argument - INTENT specified | N/A | Yes[4] | Yes[4] | N/A | Yes[4] | N/A | Yes[4] | Yes[4] | Yes[4] | Yes[4] | N/A |
| ARG5 | Illegal assignment to DO loop variable in SUBROUTINE - no INTENT specified | No | No | No | Yes | Yes | No | No | No | Yes | No | No |
| ARG6 | Illegal assignment to DO loop variable in SUBROUTINE - INTENT specified | N/A | Yes[4] | Yes[4] | N/A | Yes[4] | N/A | Yes[4] | Yes[4] | Yes[4] | Yes[4] | N/A |
| OARG1 | Illegal use of optional argument | N/A | No | Yes | N/A | Yes | N/A | Yes | No | Yes | Yes[13] | N/A |
| Source | Run-time Error | APF77 | APF95 | CVF | FTN77 | FTN95 | G77 | INTEL | LF90 | LF95 | NAS | WAT |
| ALIAS1 | Aliased dummy argument variable[14] | No | No | No | No | No | No | No | No | No | No | No |
| ALIAS2 | Aliased dummy argument array | No | No | No | No | No | No | No | No | No | No | No |
| BND1 | array bound error - X(100) | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes[13] | Yes |
| BND2 | array bound error - X(N) N is argument | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes[13] | Yes |
| BND3 | array bound error - X(N) N is in COMMON | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes[13] | Yes |
| BND4 | array bound error - X(N) N is in MODULE | N/A | Yes | Yes | N/A | Yes | N/A | Yes | Yes | Yes | Yes[13] | N/A |
| BND5 | array bound error - X(*) | No | No | No | Yes | Yes | No | No | No | Yes | Yes[13] | No |
| BND6 | array bound error - X(M:N) M and N argument - lower bound violated | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes[13] | Yes |
| BND7 | array bound error - automatic array | N/A | Yes | Yes | N/A | Yes | Yes | Yes | Yes | Yes | Yes[13] | N/A |
| BND8 | array bound error - allocatable array | N/A | Yes | Yes | N/A | Yes | N/A | Yes | Yes | Yes | Yes[13] | N/A |
| BND9 | multi-dimensional array bound error within overall array bounds | No | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes[13] | Yes |
| BND10 | array bound error - assign to actual argument which is smaller than dummy | No | No | No | Yes | Yes | No | No | No | Yes | Yes[13] | No |
| Source | Run-time Error | APF77 | APF95 | CVF | FTN77 | FTN95 | G77 | INTEL | LF90 | LF95 | NAS | WAT |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CBND1 | character bound error - local | No | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes[13] | Yes |
| CBND2 | character bound error - COMMON | No | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes[13] | Yes |
| CBND3 | character bound error - assign to actual argument which is smaller than dummy | No | No | No | Yes | Yes | No | Yes | No | Yes[4] | Yes[13] | No |
| CBND4 | character bound error - CHARACTER*(*) | No | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| UIN1 | uninitialized variable[8] - local | No | No | Yes[9] | Yes | Yes | No | Yes | Yes[9] | Yes[3] | No | No |
| UIN2 | uninitialized variable - argument | No | No | Yes[9] | Yes | Yes | No | Yes | No | Yes | No | No |
| UIN3 | uninitialized variable - COMMON | No | No | No | Yes | Yes | No | No | No | Yes | No | No |
| UIN4 | uninitialized variable - MODULE | N/A | No | No | N/A | Yes | N/A | No | No | Yes | No | N/A |
| UIN5 | uninitialized array element - local | No | No | Yes[9] | Yes | Yes | No | No | No | Yes | No | No |
| UIN6 | uninitialized array element - argument | No | No | Yes[9] | Yes | Yes | No | No | No | Yes | No | No |
| UIN7 | uninitialized array element - COMMON | No | No | No | Yes | Yes | No | No | No | Yes | No | No |
| ***Source*** | ***Run-time Error*** | ***APF77*** | ***APF95*** | ***CVF*** | ***FTN77*** | ***FTN95*** | ***G77*** | ***INTEL*** | ***LF90*** | ***LF95*** | ***NAS*** | ***WAT*** |
| UIN8 | uninitialized array element - MODULE | N/A | No | No | N/A | Yes | N/A | No | No | Yes | No | N/A |
| UIN9 | uninitialized array element - local array in SUBROUTINE | N/A | No | No | N/A | Yes | N/A | No | No | Yes | No | N/A |
| UIN10 | uninitialized array element - automatic arrays | N/A | No | No | N/A | Yes | No | Yes | No | Yes | No | N/A |
| UIN11 | uninitialized array element - allocatable arrays | N/A | No | No | N/A | Yes | N/A | Yes | No | Yes | No | N/A |
| UIN12 | uninitialized array element - saved arrays | No | No | No | Yes | Yes | No | No | No | No | No | No |
| UIN13 | uninitialized array element - INTENT(OUT) arrays | N/A | No | Yes[9] | N/A | Yes | N/A | No | No | No | No | N/A |
| DO1 | zero increment DO loop | DIV0 | DIV0 | Yes | Yes | Yes | DIV0 | DIV0 | DIV0 | Yes | DIV0 | DIV0 |
| DO2 | Illegal assignment to local DO loop variable | Yes[4] | Yes[4] | Yes[4] | Yes[4] | Yes[4] | Yes[4] | Yes[4] | Yes[4] | Yes[4] | Yes[4] | Yes[4] |
| DO3 | Illegal assignment to local DO loop variable via EQUIVALENCE | No | No | No | No | Yes[3] | No | No | No | Yes[3] | No | No |
| DO4 | Illegal assignment to DO variables in CONTAINed subprogram | N/A | No | No | N/A | Yes | N/A | No | No | Yes | No | N/A |
| ***Source*** | ***Run-time Error*** | ***APF77*** | ***APF95*** | ***CVF*** | ***FTN77*** | ***FTN95*** | ***G77*** | ***INTEL*** | ***LF90*** | ***LF95*** | ***NAS*** | ***WAT*** |
| SF1 | SUBROUTINE referenced as a FUNCTION - same file | No | No | Yes[3] | Yes | Yes | Yes | Yes[5] | Yes[3] | Yes[4] | Yes[4] | No |
| SF2 | SUBROUTINE referenced as a FUNCTION - different file[6] | No | No | No | Yes | Yes | No | Yes[5] | No | Yes | Yes[4] | No |
| FH1 | Same file opened on 2 different units | No | No | Yes | Yes | Yes | No | Yes | Yes | No | No | No |
| FMT1 | Illegal run-time format | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | No | Yes |
| CONF | Non-conformant array assignment | N/A | Yes | No | N/A | Yes | N/A | No | Yes | Yes | Yes[13] | N/A |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PTR1 | Assign via pointer after target deallocated | N/A | No[10] | No | N/A | Yes | N/A | Yes | No | Yes | No | N/A |
| PTR2 | Assign via global pointer to local array after subprogram return | N/A | No[10] | No | N/A | Yes | N/A | Yes | No | Yes | No | N/A |
| | | | | | | | | | | | | |
| UFL | underflow | Option | Option | Option | Silent | Silent | Silent | Silent | Option[11] | Option | Silent | Silent |
| OFL | overflow | Option | Option | Option | Trap | Trap | Silent | Silent | Option[11] | Option | Silent | Trap |
| DIV0 | divide zero | Option | Option | Option | Trap | Trap | Silent | Silent | Option[11] | Option | Silent | Trap |
| IOFL | integer overflow | Option | Option | Option | Trap | Trap | Silent | Silent | Option[11] | Silent | Silent | Silent |
| | Full trace-back from run-time errors | No[12] | No | Yes | Yes | Yes | No | Yes | No[12] | Yes | No | Yes |

| ***Key*** | |
|---|---|
| N/A | Not applicable to Fortran 77 compiler |
| DIV0 | Produced Divide Exception |
| Silent | No indication |
| Trap | Execution halted |
| Option | User can select Silent or Trap |

| ***Compiler Switches*** | |
|---|---|
| ***APF77*** | f77 -C -N109 %1 |
| ***APF95*** | f95 %1 -Rb -Rc -Rp -Rs |
| ***CVF*** | df %1 /check:all /fpe:0 /traceback /warn:argument_checking /automatic |
| ***FTN77*** | ftn77 %1 /undef /fullcheck |
| ***FTN95*** | ftn95 %1 /full_undef |
| ***G77*** | G77 -fno-silent -fno-ugly -Wall -W -ff90 -fbounds-check %1 |
| ***INTEL*** | ifl %1 /4Yb /Zi /d0 |
| ***LF90*** | lf90 %1 -g -chk -lst -trace -sav -pca -stchk -fullwarn -winconsole |
| ***LF95*** | lf95 %1 -chkglobal -g -co -f95 -lst -nsav -stchk -W -xref -fullwarn |
| ***NAS*** | f95 %1 -check -verbose -debug |
| ***WAT*** | wfl386 %1 -WA -DEB -BO -TR -ST -D1 |

| ***Notes*** | |
|---|---|
| | Thanks to Herman D. Knoble and Arnaud Desitter for supplying some of the diagnostic tests. You can find more tests similar to these at Hermans site: //ftp.cac.psu.edu/pub/ger/fortran/test/ |
| 1 | "Percentage Passes" is calculated from a total of 32 tests for F77 compilers, and 48 tests for F90 compilers. No score is assigned for exception handling. |
| 2 | Execution Time for TFFT.for (one of the F77 Benchmarks), compiled using the diagnostic switches (see table above) - measured on a 850 MHz Intel Pentium III with 256 Mbytes RAM, and running Windows 2000. In addition to the diagnostic switches CVF also used /link /stack:64000000. |
| 3 | Compiler issued warning message. |
| 4 | Compiler issued fatal error message - no executable produced. |
| 5 | Run-time message. |
| 6 | Arranged so that error is not visible at compile time. |

| | |
|---|---|
| 7 | Yes - if the different files are compiled at the same time. |
| 8 | A variable is "uninitialized" if it has never been assigned a value. Data elements should be initialized before their value is used. The illegal use of an uninitialized variable can sometimes be detected at compile-time, but, because it may be data and flow dependent, usually requires run-time monitoring. |
| 9 | Compile time check - may be defeated in more complex cases. |
| 10 | Page Fault. |
| 11 | Option for DOSX programs, Silent for Win32 programs. |
| 12 | Identifies execution point, but not call stack. |
| 13 | Compiler issued correct error, but did not specify line number. |
| 14 | Spotted by Forcheck. |

---

**(C) Polyhedron Software Ltd. (1998-2001)**

*Last Updated on 18/05/01*

*By John Appleyard*
*Email: john@polyhedron.com*

# Index of /pub/ger/fortran/test

| Name | Last modified | Size | Description |
|---|---|---|---|
| Parent Directory | 15-May-2001 13:46 | - | |
| ALLOC26-dt.f | 16-Mar-2001 10:01 | 1k | |
| ALLOC26-fp.f | 16-Mar-2001 10:01 | 1k | |
| ALLOC26.f | 16-Mar-2001 10:01 | 1k | |
| CON25.F | 05-Feb-2001 11:13 | 1k | |
| CON26.F | 05-Feb-2001 11:13 | 1k | |
| CON27.F | 05-Feb-2001 11:13 | 1k | |
| UIN10-DT.F | 15-Nov-2000 14:34 | 1k | |
| UIN10-FP.F | 15-Nov-2000 14:34 | 1k | |
| UIN10.F | 15-Nov-2000 14:34 | 1k | |
| UIN11-DT.F | 15-Nov-2000 14:34 | 1k | |
| UIN11-FP.F | 15-Nov-2000 14:34 | 1k | |
| UIN11.F | 15-Nov-2000 14:34 | 1k | |
| UIN12-DT.F | 15-Nov-2000 14:34 | 1k | |
| UIN12-FP.F | 15-Nov-2000 14:34 | 1k | |
| UIN12.F | 15-Nov-2000 14:34 | 1k | |
| UIN13-DT.F | 15-Nov-2000 14:34 | 1k | |
| UIN13-FP.F | 15-Nov-2000 14:34 | 1k | |
| UIN13.F | 15-Nov-2000 14:34 | 1k | |
| UIN14-DT.F | 15-Nov-2000 14:34 | 1k | |
| UIN14-FP.F | 15-Nov-2000 14:34 | 1k | |

| | | |
|---|---|---|
| UIN14.F | 15-Nov-2000 14:34 | 1k |
| UIN15-DT.F | 15-Nov-2000 14:34 | 1k |
| UIN15-FP.F | 15-Nov-2000 14:34 | 1k |
| UIN15.F | 15-Nov-2000 14:34 | 1k |
| UIN16-DT.F | 15-Nov-2000 14:34 | 1k |
| UIN16-FP.F | 15-Nov-2000 14:34 | 1k |
| UIN16.F | 15-Nov-2000 14:34 | 1k |
| UIN25-DT.F | 19-Dec-2000 14:01 | 1k |
| UIN25-FP.F | 19-Dec-2000 14:01 | 1k |
| UIN25.F | 19-Dec-2000 14:01 | 1k |
| UIN26-DT.F | 02-Feb-2001 13:08 | 1k |
| UIN26-FP.F | 02-Feb-2001 13:08 | 1k |
| UIN26.F | 02-Feb-2001 13:08 | 1k |
| UIN27-DT.F | 05-Feb-2001 11:13 | 1k |
| UIN27-FP.F | 05-Feb-2001 11:13 | 1k |
| UIN27.F | 05-Feb-2001 11:13 | 1k |
| UIN_AUXI.F | 15-Nov-2000 14:34 | 1k |
| dfdebug.bat | 26-Apr-1999 12:26 | 1k |
| lfdebug.bat | 09-Nov-2000 15:48 | 1k |
| results.txt | 18-May-2001 09:20 | 10k |
| saldebug.bat | 10-Nov-2000 09:39 | 1k |
| test1.for | 16-Feb-1999 14:23 | 1k |
| test10.for | 16-Feb-1999 14:23 | 1k |
| test11.f90 | 13-Nov-2000 10:55 | 2k |
| test12.for | 16-Feb-1999 14:23 | 1k |

| | | |
|---|---|---|
| test13.f90 | 22-Sep-1999 11:16 | 1k |
| test14.f90 | 22-Sep-1999 11:16 | 1k |
| test15.f90 | 09-Nov-2000 14:55 | 1k |
| test16.f90 | 09-Nov-2000 15:01 | 1k |
| test17.f90 | 03-Nov-2000 10:59 | 1k |
| test18.f90 | 18-May-2001 14:18 | 2k |
| test19.f90 | 28-Feb-2000 09:22 | 1k |
| test2.for | 16-Feb-1999 14:23 | 1k |
| test20.f90 | 13-Nov-2000 10:56 | 1k |
| test21.f90 | 15-Nov-2000 14:34 | 1k |
| test22.f90 | 15-Nov-2000 14:34 | 1k |
| test23.f90 | 15-Nov-2000 14:34 | 1k |
| test24.f90 | 15-Nov-2000 14:34 | 1k |
| test25.f90 | 15-Nov-2000 14:34 | 1k |
| test26.f90 | 18-May-2001 09:20 | 1k |
| test3.for | 16-Feb-1999 14:23 | 1k |
| test4.for | 16-Nov-2000 14:04 | 1k |
| test5.for | 16-Feb-1999 14:23 | 2k |
| test6.for | 26-Apr-1999 12:26 | 2k |
| test7.for | 26-Apr-1999 12:26 | 1k |
| test8.for | 26-Apr-1999 12:26 | 1k |
| test9.for | 22-Sep-1999 11:16 | 1k |

# Win32 Compiler Comparisons - Performed on an Intel Pentium III Fortran 77 Execution Time Benchmarks

| | APF77 | CVF | FTN77 | FTN95 | G77 | INTEL | LF90 | LF95 | NAS | WAT | WAT 11.0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| AC | 7.72 | 6.13 | 8.99 | 6.82 | 11.86 | 6.92 | 8.06 | 6.25 | 12.97 | 6.43 | 6.00 |
| ADI | 4.92 | 4.71 | 6.14 | 5.46 | 5.58 | 4.26 | 4.47 | 3.88 | 6.51 | 5.64 | 5.71 |
| AIR | 21.59 | 20.94 | 34.64 | 31.41 | 23.13 | 20.00 | 21.79 | 21.76 | 31.89 | 23.88 | 23.06 |
| CHESS | 24.17 | 20.56 | 28.89 | 19.42 | 19.42 | 20.10 | 20.47 | 21.86 | 38.49 | 19.93 | 19.67 |
| DODUC | 43.29 | 34.03 | 63.73 | 55.32 | 54.33 | 32.54 | 40.38 | 42.86 | 73.14 | 43.97 | 40.45 |
| LM8 | 12.81 | 8.11 | 29.25 | 12.65 | 12.70 | 12.30 | 13.71 | 12.84 | 32.96 | 13.79 | 13.79 |
| LP8 | 9.75 | 8.32 | 20.87 | 26.52 | 13.22 | 8.59 | 8.64 | 9.35 | 24.09 | 14.36 | 13.70 |
| MDB | 6.71 | 5.59 | 13.86 | 11.39 | 6.31 | 6.12 | 6.46 | 6.31 | 10.05 | 8.12 | 7.56 |
| MOLEN | 16.99 | 16.48 | 23.13 | 22.70 | 24.89 | 16.03 | 18.51 | 16.73 | 22.36 | 20.74 | 21.10 |
| PI | 14.20 | 8.38 | 9.63 | 6.34 | 32.28 | 14.18 | 15.27 | 18.92 | 7.32 | 8.08 | 7.28 |
| PNPOLY | 6.22 | 8.48 | 19.92 | 6.58 | 8.47 | 8.21 | 6.92 | 15.89 | 15.50 | 9.42 | 8.56 |
| RO | 10.26 | 17.34 | 12.66 | 12.04 | 19.89 | 9.94 | 23.67 | 10.44 | 26.71 | 16.82 | 16.80 |
| TFFT | 44.40 | 42.68 | 49.40 | 45.97 | 45.89 | 42.24 | 45.24 | 45.03 | 50.55 | 46.10 | 46.32 |
| WH4 | 21.33 | 14.58 | 31.44 | 26.41 | 36.36 | 7.09 | 15.61 | 18.68 | 40.17 | 20.90 | 20.87 |
| | | | | | | | | | | | |
| *Geometric Mean* | 13.942 | 12.393 | 20.670 | 15.908 | 18.039 | 12.065 | 14.384 | 14.404 | 22.462 | 15.072 | 14.556 |

| Compiler Switches | |
|---|---|
| APF77 | f77 %1 -O -N109 -N34 -Q100 -stack:0x800000 vms.lib |
| CVF | df %1 /fast /opt:5 /link /stack:64000000 |
| FTN77 | ftn77 %1 /opt |
| FTN95 | ftn95 %1 /optimise /p6 |
| G77 | g77 %1 -O2 -malign-double -march=pentiumpro -funroll-loops |
| INTEL | ifl %1 /G6 /O3 /Qip /Qunroll4 |
| LF90 | lf90 %1 -o3 -tpp -npca -nsav -nstchk -stack 8000000 -winconsole |
| LF95 | lf95 %1 -tpp -nchk -ntrace -nsav -nstchk -o1 -nw -stack 8000000 |
| NAS | f95 %1 -fast |

| | |
|---|---|
| ***WAT*** | wfl386 %1 /OX /5 /FP5 |
| ***WAT 11.0*** | wfl386 %1 -5 -fp5 -ox -nowarn -noerr -q |

| ***Notes*** | |
|---|---|
| | All figures are Execution Times in Seconds - measured on a 850 MHz Intel Pentium III with 256 Mbytes RAM, and running Windows 2000. Each figure is the average over at least 10 runs (much more for some). Measurement error is typically <1%. Green cells highlight figures within 10% of the fastest. Red cells indicate figures which are more than 150% of the fastest.<br><br>So far as possible, we have used the compiler switches which give the best overall results. We have not attempted to tune individual benchmarks, and, in particular cases, different switch settings may give better results. Compaq recommend /opt:4 as the norm, but suggest that /opt:5 be tried on a case-by-case basis, as it can help some programs but hurt others. Similar comments apply to other compilers.<br><br>These benchmarks are derived from an earlier set of 30. Some trivial and near-duplicate programs have been eliminated, and others have been modified to run for a reasonable time on an Intel Pentium III. Thanks to Stefan A. Deutscher for supplying the Watcom 11.0 results.<br><br>The benchmarks were also used to compare Linux compilers on the same machine.<br><br>Download Benchmark source code (F77BENCH.ZIP 114K) |

**(C) Polyhedron Software Ltd. (1998-2001)**

*Last Updated on 16/03/01*

*By John Appleyard*
*Email: john@polyhedron.com*

# Leiden University

# FORCHECK

## A FORTRAN program development aid

---

FORCHECK

- is the oldest and most comprehensive Fortran verifier on the market. It performs a full static analysis of an entire Fortran program or a separate analysis of one or more subprograms.
- detects more anomalies in your program than the compiler. Because it locates bugs as early in the development phase as possible, it saves you time and helps you to produce more reliable programs.
- is ideally suited to get a fast insight in existing and legacy programs. It composes optimal documentation with a call-tree and cross-reference tables both on the program-unit as on the program level.
- can be used as a software engineering tool in the various stages of the development process.
- can verify the conformance to the Fortran standards. Moreover it supports many language extensions of all popular compilers. FORCHECK is fully configurable so you can tune the analysis and output to your needs.
- can store the global information of the analyzed program-units in libraries. You can reference these libraries in subsequent FORCHECK runs to verify the consistency of all references and common-blocks.
- is very suitable as a cross-platform development tool.
- is available on many platforms from PC to supercomputer. It supports most Fortran extensions of all popular compilers.
- is sold with full guarantee and support.
- is in use at many sites.

Follow this link to Leiden University's Web Site.

Click here for Forcheck price list

# Win32 Fortran Compiler Comparisons

## Third Party Packages

| Package | APF | CVF | FTN77 | FTN95 | G77 | INTEL | LF90 | LF95 | NAS | WAT |
|---|---|---|---|---|---|---|---|---|---|---|
| *INTERACTER* | No | Yes | Yes | Yes | Yes | No | Yes | Yes | No | Yes |
| Winteracter | Yes | Yes | No | Yes | No | No | Yes | Yes | No | No |
| GINO | Yes | Yes | Yes | Yes | No | No | Yes | Yes | ? | ? |
| GINOMENU | Yes | Yes | Yes | Yes | No | No | Yes | Yes | ? | ? |
| IMSL | Yes | Yes | Yes | Yes | ? | ? | Yes | Yes | ? | ? |
| Nag libraries | Yes | Yes | Yes | Yes | ? | Yes | Yes | Yes | ? | ? |
| *plus***FORT** | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| Visual KAP | Yes | Yes | No | No | No | No | No | No | No | No |
| RealWin | Yes | Yes | No | No | No | No | Yes | Yes | No | No |
| f90SQL | Yes | Yes | No | Yes | No | No | Yes | Yes | No | No |
| Forcheck | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| f90gl | Yes | Yes | No | No | No | ? | Yes | Yes | Yes | No |

**(C) Polyhedron Software Ltd. (1998-2001)**

*Last Updated on 15/03/01*

*By John Appleyard*
*Email: john@polyhedron.com*

# *Winteracter, X/Winteracter and INTERACTER* from ISS Ltd


| | | | | |
|---|---|---|---|---|
| ***Winteracter*** | News | Pictures | Demos | ***X/Winteracter*** |
| FAQ | Visual Tools | Subroutines | Price List | ***INTERACTER*** |

## Recent news items :

- ***Winteracter*** Version 3.0 *November 24th 2000*
- ***Winteracter*** v2.30f Update *October 23rd 2000*
- ***X/Winteracter*** *for UNIX/Linux (June 2000)*
- ***Winteracter*** Version 2.30 *(May 26th 2000)*
- ***INTERACTER*** Version 5.0 *27/Oct/2000*
- ***INTERACTER*** for Alpha Linux Compaq Fortran 95 *24/Feb/2000*
- ***INTERACTER*** Sunsoft Fortran 90 v2 *10/Feb/2000*
- ***Winteracter*** Version 2.20 released (*30th Nov 1999*)
- ***Winteracter*** Version 2.10 released (*28th May 1999*)
- ***INTERACTER Express*** *(1st September 1999)*
- ***INTERACTER*** for Fujitsu Fortran 95 on Linux (*August 6th 1999*)
- ***INTERACTER*** for SPARC Based Linux Systems (*16th April 1999*)
- ***INTERACTER*** for Alpha Based Linux Systems (*16th April 1999*)
- ***INTERACTER*** Version 4.20 released (*30th October 1998*)

***Winteracter*** is for Fortran 90 users on the Win32 platform. Programmers can create true Windows "look and feel" programs with ***Winteracter*** without resorting to API access, mixed language programming or DLL's. ***Winteracter*** provides:-

- Visual tools for user interface design
- Comprehensive subroutine library
- Modern Fortran 90 oriented programming interface
- Mature platform/device-independent graphics

***X/Winteracter*** is a major new implementation of ISS's W*interacter* GUI toolset, designed for use with X Windows - the standard Unix/Linux windowing system. X/W*interacter* is based on the Motif toolkit and offers a high degree of compatibility with the Win32 implementation.

***INTERACTER*** is ISS's multi-platform product, enabling interactive Fortran 77/90 based programs to run under DOS, Windows, Unix and VMS.

Like ***Winteracter***, ***INTERACTER*** consists of a subroutine library and a complementary set of utility programs. ***INTERACTER*** employs a generic look + feel across all platforms, using a different calling interface to ***Winteracter***. However, both products provide the same graphics and o.s. interface routines. V5.00 will be
the last INTERACTER release. ISS will continue to offer full technical support, fix bugs and support new compiler releases. However, ISS intend
to focus their development resources on ***Winteracter*** and related products in future.

INTERACTER provides:-

- Comprehensive subroutine library
- ***INTERACTER Toolkit*** (form designer, on-line help, etc.)
- Same platform/device-independent graphics as Winteracter

***INTERACTER* and W*interacter* are produced by Interactive Software Services Ltd.**

# GINO PRODUCT SUMMARY

GINO is a range of portable graphics and GUI development tools offering the programmer everything from simple line-drawings to complex fully-interactive three-dimensional diagrams.

## New! GINO v5.0

- **143 new routines**
- **Re-written and extensively illustrated User manual**
- **New 'Getting Started' Guide**
- **New Reference Cards**
- **Complete on-line documentation in Help and PDF format**
- **Single and Double Precision versions on CD**

**Fortran Libraries:**

| | |
|---|---|
| **GINO-F** | GINO-F is an established Fortran subroutine library providing a complete range of low and high level routines giving complete control over every aspect of a graphics application. |
| **GINOMENU** | GINOMENU is a Fortran subroutine toolkit for developing GUI applications across all major platforms. It provides extensive window and widget building modules as well as access to all the high and low level graphics facilities featured in GINO-F, GINOGRAF and GINOSURF. |
| * **GINOGRAF** | GINOGRAF is an extensive Fortran subroutine library providing a comprehensive range of technical graph drawing facilities. |
| * **GINOSURF** | GINOSURF is a sophisticated contour and surface drawing package enabling the programmer to obtain comprehensive three and four dimensional data visualisation. |
| * **CALGINO** | CALGINO is an interface library that converts Calcomp type calls to GINO-F calls, enabling existing applications to be output to a much wider variety of output devices. |

*** needs GINO-F**

**GUI Builder:**

| | |
|---|---|
| **GINOMENU Studio** | GINOMENU Studio is an interactive GUI builder and development environment for the Fortran programmer. Developers can now create a graphical user interface by drag-and-drop methodology without having to write any source code. |

**Fortran Converters:**

| | |
|---|---|
| **MENUGEN** | MENUGEN is a converter that translates Visual Basic GUI form-text files to GINOMENU Fortran source. |

**C Products:**

| | |
|---|---|
| **GINO-C Bundle** | The GINO-C Bundle is a set of libraries comprising C versions of GINO-F, GINOGRAF and GINOSURF as described above. |

Platform availability

Prices

GINO Products are available from Polyhedron Software Ltd, Linden House, 93 High Street, Standlake, Witney, Oxon, OX8 7RH, United Kingdom.

Tel +44 (0)1865 300579, Fax +44(0)1865 300232, email wwwsales@polyhedron.com

---

# GINOMENU

---

## Product Description

GINOMENU is a Fortran subroutine toolkit for developing GUI applications that are totally platform independent and can be ported from Windows to DOS to UNIX without the need for any change of code or running of any converters. It provides extensive window and widget building modules allowing professional user-interfaces to be created under Windows with direct access to the Win16 or Win32 API. Alternatively, by using its unique built-in emulation mode, user interfaces can be built under DOS, UNIX or OpenVMS providing applications with an authentic PC-style or Workstation-style look and feel.

## On-the-Fly Widget Creation

Another big advantage of GINOMENU is that all widget creation is done within the Fortran program. This allows ease of maintenance for one thing, but more importantly allows the application to generate widgets on-the-fly. Resources can easily get used up under Windows and there is no point creating dialog boxes that are not going to be used. GINOMENU applications can delay the creation of widgets until they are actually required and they can be created depending on end-user influences rather than the developers.

## Features

Available widgets include panels, buttons, sliders, toggles, radio boxes, text-entry fields, numerical-entry fields and graphics frames. Window widgets include Complex Dialogue boxes, a file browser and pop-up/pull-down menus. 100 icons are supplied with the package in two different sizes and access to BMP files and .ICO files is provided for including standard Windows icons within an application. Attribute control includes accelerator keys, size and type of slider, size and type of toggle, type of icon, colour definition, titling etc. Various interaction methods can be utilized including graphics expose, window and widget callbacks, iconization, selection and re-size.

## Graphics

GINOMENU not only provides GUI building modules, but also includes basic graphics routines to enhance the look of the application. However even more power can be realized when the optional high level graphics facilities featured in GINO-F, GINOGRAF and GINOSURF are added to provide a complete graphics development environment for any application.

## Ease of Use

GINOMENU is unique in not only giving a programmable GUI system to Fortran users but making it extremely easy to use; No knowledge of system event loops or window handles is required, no complicated mixed-language calls, no include files are necessary and no knowledge of Resource files or Resource Compilers is required. On-line context-sensitive help is supplied to speed up the learning process and a basic program containing a simple window can be created with less than 16 lines of code!

The extensive functionality, ease-of-use and flexibility of GINOMENU provides the programmer with a powerful development toolkit for generating a fully portable GUI and graphical display system.

# Highlights

- Create true Windows GUI apps.
- Create apps that are fully portable to DOS, UNIX & OpenVMS
- Fully responsive methodology allowing widget creation on-the-fly
- Basic graphics built-in
- Fully integrated with the other GINO products for extended 2D and 3D graphics
- Very easy to use
- Compatible with Fortran 77 and 90
- No mixed language calls
- No Resource files required
- Hundreds of widget types & styles
- Built in icons and access to BMP/ICO files

If you already have Visual Basic, GINOMENU interface design is now even easier with MENUGEN, a VB to GINOMENU Fortran Source converter

---

# The IMSL Fortran 90 MP Mathematics and Statistics Libraries (F90 MP)

This new generation of Fortran 90-based algorithms are optimized for multiprocessor and other high-performance systems:

- Operators and function modules that provide faster and more natural programming through an object-oriented approach.
- Native Fortran 90-coded, *not translated*, subroutines designed to fully optimize Fortran 90 features for efficient program execution.
- The world-renowned IMSL FORTRAN 77 Mathematics and Statistics Library included.
- Programmers can leverage available CPUs, regardless of type, to maximize hardware investments and optimize Fortran application performance.

F90 MP provides a dynamic interface for computing mathematical solutions over a distributed system via Message Passing Interface (MPI). Routines offer a simple, reliable user interface.

F90 MP's MPI-enhanced interface provides:

- Computational control of the server node.
- Scalability of computational resources.
- Automatic processor prioritization.
- Self-scheduling algorithm to keep processors continuously active.
- Box data type application.
- Computational integrity.
- Dynamic error processing.
- Homogeneous and heterogeneous network functionality.
- Use of descriptive names.
- Short, required argument lists.
- Packaged user-interface blocks for the IMSL F90 MP Library.
- Interface blocks for the entire IMSL FORTRAN 77 Library.
- A suite of testing and benchmark software.

## The IMSL FORTRAN 77 Mathematics and Statistics Libraries (FNL)

FNL is comprehensive resource of more than 900 FORTRAN 77 subroutines for use in general applied mathematics and statistical data analysis and presentation in scientific and business applications.

- Version 3.0 includes more efficient online documentation and enhancements to linear algebra, optimization and linear equations.
- Routines and interfaces are consistent over 65 supported computer platforms.

The strength and precision of the IMSL Numerical Libraries have been evolving steadily since 1970. Each subroutine and algorithm has undergone rigorous testing and quality assurance.

Modules in the IMSL® Math Libraries (link to Visual Numerics Inc)

# NAG Fortran Library

---

## *Numerics and Statistics ...*

## Development

NAG® has a worldwide reputation for the excellence of its Fortran Library which contains over 1000 user-callable routines.

NAG has released a library in the new standard, Fortran 90, and a library of parallel algorithms in Fortran for distributed-memory machines. This pioneering work ensures that NAG will continue to stay at the forefront of numerical software technology.

---

## Contents

The Fortran Library is organized into chapters, each chapter devoted to a branch of numerical or statistical computation. A full list of the routines is available but some of the areas covered include:

**Minimization**

- constrained or unconstrained
- with or without derivatives
- nonlinear least-squares
- linear and quadratic programming (LP and QP)
- mixed integer linear programming (ILP)
- nonlinear programming

**Ordinary Differential Equations**

- initial value problems
- boundary value problems
- differential alegbraic problems
- Sturm-Liouville problems

**Partial Differential Equations**

- elliptic equations
- Helmholtz equations
- systems of time-dependent PDEs in one space dimension using the method of lines, first and

second order

**Transforms**

- FFTs for single or multiple series of real, complex or Hermitian data
- sine and cosine transforms
- convolutions
- inverse Laplace transforms

**Quadrature**

- fully adaptive with specialised weight functions over finite and infinite intervals
- multidimensional quadrature
- Gaussian quadrature

**Linear Algebra**

- linear systems, (real, complex, dense, banded, tridiagonal, Toeplitz, sparse)
- linear least-squares (dense or sparse)
- singular value decomposition
- eigenvalue problems, real or complex, symmetric or nonsymmetric
- generalized eigenvalue problems
- iteratrive solvers and preconditioners for large sparse linear systems
- high performance software derived from LAPACK
- Basic Linear Algebra Subprograms (BLAS)

**Nonlinear Equations**

- root of single equation
- solution of general system
- roots of polynomials

**Integral Equations**

- Fredholm nonsingular, 2nd kind
- Volterra convolution
- weakly singular convolution Volterra-Abel

**Curve and Surface Fitting**

- interpolation/approximation
- one or two dimensions (regular or scattered data)
- polynomials or splines
- least-squares, $l_1$ or $l\alpha$ fits

**Special Functions**

- Bessel, Hankel and Airy functions, real and complex arguments

- Kelvin functions
- elliptic, exponential, trigonometric and Fresnel integrals
- Gamma and related functions

**Basic Statistics**

- data summaries and displays
- distribution functions

**Random Number Generation**

- wide range of distributions
- time series
- matrices
- random sampling

**Correlation and Regression**

- product-moment correlation
- non-parametric correlation
- simple linear regression
- multiple linear regression (general linear model)
- generalized linear models
- robust techniques

**Multivariate Methods**

- principal component analysis
- canonical correlation analysis
- factor analysis
- cluster analysis
- discriminant analysis
- mutlidimensional scaling

**Analysis of Variance**

- factorial designs
- general block designs

**Univariate Estimation**

- confidence intervals
- maximum likelihood estimation
- robust estimation

**Nonparametric Statistics**

- many standard tests

- goodness of fit tests
- tests of randomness

**Smoothing**

- smoothing splines
- density estimation

**Contingency Table Analysis**

- $X^2$ statistics
- multiway tables
- latent variable model

**Survival Analysis**

- Kaplin Meier estimates
- Cox's regression model

**Time Series Analysis**

- ARIMA model fitting and forecasting
- multivariate model fitting and forecasting
- spectral analysis
- Kalman filters

**Sorting**

- integer, real or character data

**Utilities**

- machine and mathematical constants
- printing procedures
- date and time
- CPU time

---

# The Benefits

**Accuracy and Reliability**

The NAG Fortran Library has been developed in both single and double precision. On many systems both precisions are provided. On other systems the precision chosen is that which is considered most suitable in general for numerical computation (usually double).

The correctness of each Library routine is evaluated and verified by specially written test programs which are performed on each of the machine ranges for which the Library is available. Only when an implementation satisfies NAG's stringent accuracy standards is it released.

**Cost Effective**

The NAG Fortran Library, with over 1100 user-callable routines, gives easy access to complex and highly sophisticated algorithms. At less than the cost of employing a programmer for a month, it is the most cost effective way of solving your numerical and statistical problems.

**Saves Development Time**

Developing robust, reliable and accurate code to solve the problems that the NAG Fortran Library can tackle takes many years of effort. Solving your problems with the NAG Fortran Library saves development time and money. Why wait longer for your results?

**Protects your Investment**

You invest much time and money in your application programs. Once they are installed, you rely on their results. NAG routines are maintained on a regular basis, fully documented and supported and designed for portability. Whatever the future holds, you can rely on NAG for your continuing numerical and statistical software needs.

---

# Software

For each target platform the software supplied includes the compiled library, example program material and documentation for all available routines. Each example program, together with a data file (if needed) and a results file, illustrates the use of a particular routine. It can also be used as a template for your own application.

To help your staff install NAG software on your hardware, a detailed Installers' Note is provided. A separate Users' Note gives implementation-specific details that may be required by the end-user.

NAG software is distributed on a range of media, to suit your requirements. Full details of these media options can be supplied on request.

To assist users who wish to call the Fortran Library from C programs, a complete set of C Header Files is available.

---

# Documentation

When you purchase the NAG Fortran Library you are automatically sent one copy of the NAG Fortran Library Manual or , if you are a PC user, a form of the On-line Browser.

The Manual is the principal printed form of documentation for the Library. It has the same chapter structure as the Library: each chapter of routines in the Library has a corresponding chapter (of the same name) in the Manual. General introductory documents and indexes can be found at the beginning of the Manual.

Each Chapter begins with an Introduction, giving a general background to the types of problem that can be solved, and guidance on selecting the most suitable routines. This is followed by a list of Contents and then a routine document for each documented routine in the chapter.

The On-line Browser supplied instead of the printed manual for PC users, is based on the On-line Information Supplement, described later in this article. DOS users receive a simple mechanism for viewing the On-line Information Supplement files; for users of the 16 and 32 bit DLLs this capability is augmented by the facility to cut and paste appropriate Visual Basic Declare statements and C Header File information into user programs.

The library documentation is also available in hypertext-based form as one of our TextWare products. This is described more fully later in this article.

---

# Support Service

Full product support is included in the annual licence fee and provides you with technical support (via NAG's Response Centre). You will also receive a news-sheet, called Network, and Maintenance Bulletins. New releases of the library are automatically sent to you as and when they become available. Clients who prefer to purchase a perpetual use licence can purchase full NAG support for 18% of the annual licence fee.

**Technical Support**

Technical Support is available via the NAG Response Centre, where details of your call will be taken and a log number (a unique reference to identify your query within NAG) will be given. Your details will be passed on to the relevant experts within NAG who can then fully investigate your problem or query before making a call back to you. You can be assured that the answers you receive will have been thoroughly researched ... usually by the very professionals involved in the development of the product. The NAG Response Centre can also be used just as a general enquiry service.

**Network**

Network is a periodical news-sheet which contains articles on new products and enhancements to existing products. Any events where you can meet NAG personnel and learn more about our products and services are listed in Network.

Your copy of Network also tells you about any special offers which may be available on NAG Products.

**New Releases and Maintenance Bulletins**

New releases are issued on a regular basis, primarily because of enhanced functionality, but also to include any necessary error corrections. Documentation updates are supplied with each new software release.

Maintenance Bulletins are issued periodically, if appropriate, between releases of the library. They notify users of any NAG Fortran Library routines with confirmed errors and give details of the necessary corrections.

**The NAG WWW Service**

The NAG website provides information about all of NAG's current products and services. In

addition, it gives access to technical reports, summaries and code, as well as links to other sites of interest.

For the Fortran Library, a full contents list is available, with search facilities based either on keywords or the GAMS classification system. On-line documentation and example programs are also available for all the routines.

To fully explore these and other benefits view our whole website.

**NAG Users Association**

When you purchase your NAG Fortran Library you will, no doubt, want to become an active member of the NAG Users Association. As well as providing a valuable forum for the exchange of ideas, NAGUA hold regular meetings and seminars to help you get the most out of your NAG products.

Your membership of NAGUA ensures that NAG remains in touch with the changing requirements of its many users.

---

# Specialist Libraries

Specialist numerical libraries in Fortran are available separately for users with more intensive requirements: the Data Approximation Subroutine Library (DASL), developed at the National Physical Laboratory, for data-fitting problems; and the Harwell Sparse Matrix Library (HSML), extracted from the renowned Harwell Subroutine Library, for problems involving sparse matrices (real and complex linear systems, eigenvalue problems, nonlinear equations, approximation of derivatives and optimisation).

---

# Availability

The NAG Fortran Library is available on a very wide range of computer systems. For each distinct system, an implementation of the Library is specially prepared by NAG. This implementation is then distributed to clients as a fully tested and compiled library.

The library is also provided as a set of DLLs (Dynamic Link Libraries) for use on Windows systems. In this form it can be called from Visual Basic, Excel or Visual C++. Visual Basic Declare Statements and C Header Files are included.

For Fortran 90 programmers, the Library is provided alongside our new Fortran 90 Library, *fl90*, as part of our FL90plus product. Fortran 90 interface blocks are included for all routines, and routines from both libraries can be called from the same program. The new *fl90* library offers a more attractive interface, taking advantage of new features of Fortran 90, but does not yet match the full functionality of the Fortran 77 Library.

For full details of the implementations available, please contact NAG.

---

# *Products to Supplement the Library*

## TextWare

Nag has recently launched a new range of electronic hypertext-based documentation called TextWare.

For the Fortran Library, TextWare is currently available for use on a range of Unix platforms. The product consists of an electronic book form of the full manual (apart from the indexes) together with a browser, and uses the DynaText electronic publishing system developed by Electronic Book Technologies Inc.

Its features include:

- Typeset Mathematics
- Pop-up windows
- Table of Contents Navigation
- Hypertext Navigation
- Searching
  - Context searches
  - Boolean searches
  - Partial searches
  - Proximity searches
- Copying and Exporting of text
- Graphics
- Alternative Views.

# *plus*FORT Version 6

Polyhedron's toolkit for Fortran programmers.

- SPAG - Fortran source code restructuring
- SPAG - Questions and Answers
- Quality Assurance using *plus***FORT**
- GXCHK - Global Static Analysis
- Dynamic Analysis
- Coverage Analysis
- AUTOMAKE - automated make for Fortran and C
- QMERGE - Version Selection
- *plus***FORT** for Linux
- Distributors

# Visual KAP

Visual KAP is engineered to make your life easier -- to be a preprocessor that allows Windows NT Fortran 77/90/95 and C programmers to get significant speedups. Visual KAP automatically parallelizes your code by inserting calls to its parallel library to allow your code to run on multiple processors.

Visual KAP's advanced optimization techniques also produce the most efficient serial code possible, allowing you to confirm execution efficiency.

Visual KAP accepts standard Fortran 77/90/95 and ANSI C source code, runs on all Pentium classed processors under Windows 95/98/NT/2000, and targets the Compaq Visual Fortran compiler. It is also available for the Absoft Fortran compiler.

Visual KAP is available as a **serial version** for single processor optimizations, **parallel version** which is available for multi-processor optimizations, and **OpenMP** - contains all features of the Autoparallel Version and processes OpenMP directives into calls to the included parallel run-time library for parallel execution on 1 to 2 processors, or unlimited processors (according to the licence type).

## What is OpenMP?

Kuck & Associates, in conjunction with major hardware and software vendors, has launched the OpenMP specification for programming shared memory multiprocessor computers. OpenMP specifies a portable and simple to use directive-based parallelization model with several advantages over other approaches:

- Multi-platform, multi-OS support.
- Incremental parallelization of sequential codes.
- Single source code solution.
- Easier to use than threads or message passing.
- Allows verifiable correctness of parallel programs.

## What is the difference between Visual KAP Serial Version and Parallel Version?

Visual KAP Serial Version is for use on single processor computers. Serial Version performs serial optimizations via code restructuring to speed up sequential programs. Serial Version is available for Windows 2000, Windows NT 4.0 and Windows 95.

Visual KAP Parallel Version produces code for use on multiprocessor computers. Parallel Version automatically converts a serial program for parallel execution on one or more processors. Parallel Version does not support parallel directives. For parallel directive support see the KAP/Pro Toolset for a description of KAI Software's parallel programming tools.

Parallel Version includes all the serial optimizations of the Serial Version. Parallel Version is available for Windows 2000 and Windows NT.

## What is the difference between Visual KAP Parallel Version and Visual KAP for OpenMP?

The Visual KAP Parallel Version produces code for use on multiprocessor computers. The Parallel Version automatically converts a serial program for parallel execution on one or more processors. The Parallel Version does not support parallel directives.

Visual KAP for OpenMP also automatically converts a serial program for parallel execution on one or more processors. In addition, Visual KAP for OpenMP can automatically insert and process OpenMP parallel directives. If you have an existing application that does not contain OpenMP directives, you can use Visual KAP for OpenMP to automatically insert OpenMP directives in your code. This allows you to examine or modify the OpenMP directives for maximum control over your program's parallel execution. If you have an existing application that already contains OpenMP directives, you can use Visual KAP for OpenMP to process the OpenMP directives into calls to the included parallel run-time library. The resulting code can then be compiled to create an executable capable of running in parallel.

The Visual KAP Parallel Version supports C and Fortran. Visual KAP for OpenMP currently only supports Fortran.

# Visual KAP Features

Visual KAP is an easy-to-use precompiler for Fortran 77/90/95 and C. By precompiling with Visual KAP you can automatically parallelize and optimize programs for all Pentium class processor systems using Windows 2000/NT/95 and Compaq Visual Fortran.

# What Visual KAP Does

Visual KAP examines each of your ANSI-standard C or Fortran source files, and generates a new source file which performs the same actions and calculations, but is optimized in various ways to run faster. On multiprocessor systems, Visual KAP automatically splits the code, where possible, into multiple parallel threads of execution, often resulting in dramatic speed increases (since Windows 95 does not support parallelism, parallelism optimizations are available only under Windows NT/2000).

On single-processor systems, Visual KAP can still be used as an optimization validator, to verify that your code is optimized. Its state-of-the-art scalar optimization algorithms can detect portions of the code which are not optimized, and optimize them to run faster.

Visual KAP combines a decade of optimizing technology by Kuck & Associates, Inc. with a simple and intuitive graphical user interface. Visual KAP works with optimizing compilers to provide additional speedups beyond what the compiler's built-in optimizer provides.

# Simple Graphical User Interface

Visual KAP has a highly intuitive graphical user interface, with support for a variety of powerful features, including:

- Single-click processing of multiple files
- Easy graphical access to Visual KAP's most powerful options
- Informative status display during and after optimization
- Comprehensive on-line help

For a graphical view of the Visual KAP user interface, see the Visual KAP user interface.

# Powerful Command-line Interface

For users who prefer to access Visual KAP's powerful optimizations through a command line or from a Makefile, Visual KAP offers a stand-alone command-line executable. The command line version also offers far more options than the graphical interface-- nearly 100 different optimization options are available.

# System Requirements

Visual KAP requires:

- All Pentium class processor systems running Windows 95/98, Windows NT 4.0 or Windows 2000
- 16 Megabytes of RAM (32 Meg preferred)
- 4 Megabytes of hard drive storage space
- Visual KAP is currently supported by the Compaq Visual Fortran compiler, the Microsoft Fortran PowerStation compiler, the Absoft compiler and the Microsoft Visual C++ compiler.

NOTES: Visual KAP supports the ANSI Fortran 95 standard, and numerous industry extensions to that standard, including VAX, DoD, and CRAY Fortran extensions. There are vendor-specific extensions that are not currently supported, including some Microsoft Fortran extensions.

Prices: Contact details: How to order

INDOWSWAY
Do Windows with Fortran

BROCHURE
PRODUCT DEMO
CUSTOMIZE
COMPANY NEWS
PRICE/ORDER
SUPPORT
MORE ABOUT US

Home

## *Windows and Fortran*

Welcome to Windows programming! If you have never written an application for Windows before and you've heard it is complicated, RealWin will eliminate your fears. With RealWin you can transform your DOS-hosted Fortran program into a great Windows application. When you're done, your whole user interface is coded in Fortran.

RealWin is a custom library designed in a partnership with some pretty demanding clients. The powerful features that our custom clients use are available at a "shrinkwrap" price. You can order your own special features or just use the existing ones.

RealWin lets a Fortran programmer create full-featured applications for Microsoft 32-bit Windows platforms. Most of the details of handling Windows messages, menus, dialog boxes, etc., are done automatically within RealWin routines. With RealWin doing the hard work, you only have to consider the high-level aspects of your user interface. All of RealWin's functions, subroutines, parameter definitions and interfaces are in a Fortran 90 module. RealWin procedures typically have many optional arguments which means you can use default values to get your program written quickly and then go back later and customize. (See sample screen)

If you want to create applications for Microsoft Windows with Fortran, then you should be using RealWin. Just call RealWin's powerful Fortran interface routines to get access to the Windows API. Don't get bogged down with mixed-language programming. Don't settle for some wimpy emulation of Windows that

## RealWin Features

See All Functions

### Fortran 90

RealWin takes advantage of key features of Fortran 90 so that your code is easy to read and easy to enhance. You don't have to change any of your FORTRAN 77 code in order to use RealWin. Any Fortran 90 code you must write can be cloned and modified from the RealWin example programs.

### Menus

Menu bar, drop-down menus, cascading menus, dividers, menu help, accelerators, mnemonics, keyboard interface, enable, disable, and graying.

### Status Bar

A variety of ways to display date & time, caps lock, scroll lock, num lock, program defined messages, line & column for scrolling box, proportional spacing, can all be changed on the fly by your program.

### Tool Bar

Built-in or user-defined buttons, button sizing, tooltip help. Program can disable, enable, hide, and show buttons on the fly.

### Windows and Boxes

A multitude of ways to customize your display. "Boxes" can be used to divide up a window. Select size, position, text color, background color, title bar, font, cursor, icon, and many other appearance features. Automatically scroll a text file. Use automatic screen painting or Fortran paint subroutine.

### Dialog Boxes

Modal or modeless, push-buttons, radio buttons,

won't let your application evolve. RealWin is the way to get from Fortran to Windows. (See complete list of functions)

## Real Win is the Answer

- **Look and operation are real Windows!**
- **Code exclusively in Fortran**
- **Capitalize on Fortran 90 features**
- **Many optional arguments with defaults to get started quickly, then enhance**
- **System constants are defined in a Module**
- **Interfaces to routines are all explicit, so compiler checks them**
- **Locations can be proportional by addressing with real numbers**
- **Resizing logic is automatic**
- **You can order new features according to your specification!**

## What Others are Saying

*"I have already quickly ported the DOS-based programs to Windows and I am pleased with the speed and ease of that effort. I have also selectively added dialog boxes for input and have added a status bar and menus ..."*
**T.J. Blattner**
**Solar Turbines Inc. - a Caterpillar Company**

*"RealWin has greatly simplified the task of upgrading my Rail Traffic Controller program to be a Windows based application. I spend the majority of my time programming the crux of the application rather than worrying about the cosmetic Windows interface details. RealWin completely hides the often nasty and arcane details of Windows programming through a series of easy to understand routines. RealWin contains access to all the*

check boxes, edit boxes, list boxes, and combo boxes. Default positioning of controls, Call Fortran routine when a button is pushed. Common Dialog boxes for color, font, print, print setup, open file, and save file.

### Colors
16 predefined colors, user-definable colors via Color Dialog Box, or create any color with the RGB function.

### Graphics
Create brushes and pens, specifying color, width and pattern. Draw lines, pixels, ellipses, arcs, chords, rectangles, bitmaps, polygons, bezier curves, and polylines.

### Text
Programmer or user can select any Windows font, specifying point size, italic, rotation, etc. Write text in a variety of ways including automatic formatting.

### Printing
Use the screen graphics routines to write to the printer. Place images wherever you want on the paper. Use the Print dialog box and Page setup dialog box to let the user tell you what he wants.

### Mouse
Capture any mouse operations you want. Automatic rubber-band box and line.

### Miscellaneous
Many conversion, inquiry and utility routines, such as: get_rect, x_pixel_to_inches, lower_case, upper_case, message_box, delete_object, get_volume_info, copy_file, delete_file, get_device_capabilities, load_cursor, load_bitmap, load_icon, and set_foreground.

### User Manual
Indowsway provides a 100+ page manual with RealWin. The manual provides plenty of

*Win32 calls necessary for a professional interface. After studying a few examples, I was up and running in Windows in a matter of hours.*
*To me, the amazing part of RealWin is how little code is required to write an interface. The combination of RealWin with Fortran 90 provides developers with a powerful language and a complete interface package. I highly recommend RealWin for industrial strength applications!"*
**Eric Wilson**
**Berkeley Simulation Software**

*"I'd like to extend a heart-felt 'THANK YOU' for developing RealWIN. It's really made API/GUI coding tasks far more convenient and much easier!!! It's also tied ALOT of loose ends when it comes to API confusion."*
**Todd M. Sherman**
**"U.S. Government"**

*"My experience with beta version is more than positive. The basic idea is very good. It is easy to include RealWin in existing application with small changes in program structure. The sample cases are more than enough to start the usage of the library. Until now I have not experienced any problem with RealWin usage."*
**Davor Grgic**
**Faculty of Electrical Engineering and Computing Zagreb**

*"Keep up the good work; I think RealWin is a terrific product."*
**Dr. Alan Minchinton**
**Research Associate**
**ICI Explosives**

**System Requirements:**

examples and sample code to get you up and running as quickly as possible.
**See All Functions**

**Price**
**RealWin (all Editions)**
**$395**

- Lahey's LF90 version 3.0 or above or,
- Lahey's LF95 version 5.0 or above or,
- Compaq (Digital) Visual Fortran version 5.0 or above or,
- Absoft Fortran 90,

and one of the following:

- Windows 98
- Windows 95
- Windows 2000
- Windows NT version 4.0 or above

# f90SQL by Canaima Software

f90SQL is a library of functions and subroutines that work as an interface between your Fortran programs and Microsoft Windows Open Database Connectivity (ODBC) API. f90SQL offers a convenient and familiar way to directly read and write data from your Fortran programs to many applications formats. The list below includes a few of the most common formats, but your options are basically unlimited. As long as the application offers an ODBC interface to its data files (almost all DBMS in the market today do), using f90SQL you can read/write data to the native application's format directly from your Fortran programs.

### Common Application Formats Accessible Through f90SQL

- Excel
- Lotus 1-2-3 spreadsheets
- Microsoft Access
- FoxPro
- Paradox
- Oracle
- Ingres
- Informix
- Microsoft SQL-server
- Any application format that offers an ODBC interface to its data files

### Key Benefits

- Use SQL language to access application data, regardless of the native format of the data.
- f90SQL offers an interface to all standard ODBC 3.5 API functions.
- Your Fortran programs get direct access to the application's data. No more clumsy, time consuming, error prone extractions to ASCII files or importing data generated from your programs into the application.
- f90SQL subroutines use the familiar Fortran protocol to receive and return parameters. You do not have to deal with DLLs or the ODBC-API calls and parameters directly.
- f90SQL is supported by Canaima Software, a consulting company with ample experience in scientific and database programming.

f90SQL supports Digital Visual Fortran versions 5 and 6, Absoft Pro Fortran, Lahey LF95, Lahey Fortran 90, and Salford FTN95. f90SQL makes extensive use of Fortran 90 language features, and is not available for Fortran 77 compilers.

Frequently asked questions

### Pricing and support

f90SQL-light is free for certain users and uses (check the f90SQL-light license agreement to see if you

qualify). The license agreement for f90SQL-light does include support, but many of your questions are likely to be answered in the f90SQL discussion group:-
http://www.canaimasoft.com/f90sql/Discussion_welc.htm

Download f90SQL-light

If you do not qualify for f90SQL-light or you want the functionality of f90SQL-pro (including the new f90ADO when released), you can buy a license for f90SQL-pro.

The f90SQL-pro license includes:

- f90SQL-pro library
- An electronic version of the f90SQL-pro User's manual (and the documentation for f90ADO when released)
- Unlimited 6 months e-mail support and 1 year updates.

You can also buy multi-user licenses and pre-paid phone support time.

Printed versions of the manuals and documentation for f90SQL and f90ADO are also available at cost price.

To order - Contact sales@polyhedron.com

# f90ADO

### Product Description

f90ADO is the latest product from Canaima Software. f90ADO is a library of functions and subroutines that give your Fortran programs access to the rich and simple collection of ActiveX Data Objects (ADO), the last standard for Microsoft Windows database connectivity. ADO is an object oriented interface to data in native format. It allows access to ODBC and OLE-DB databases.

f90ADO is currently in alpha-testing.

### Key Benefits

- Powerful, optimized data access capabilities, but still easy to use.
- Your Fortran programs get access to the latest database connectivity technology, including ODBC and OLE-DB.
- Standard interface, regardless of the native format of the data.
- Use the same standard commands, regardless of the programming language your are working on. Develop in Fortran, C++, Visual Basic or ASP using the same logic and access structure in all languages.
- Supported by Canaima Software, a consulting company with ample experience in scientific and database programming.

Reserve your copy - contact sales@polyhedron.com

# f90gl

[Documentation]... [Software]... [News]... [Related Links]

---

f90gl is a public domain implementation of the official Fortran 90 bindings for OpenGL. The current version is Version 1.2.4 This version implements the interface for OpenGL 1.2, GLU 1.2, GLUT 3.7, and several extensions. It supports several Unix workstations and Windows 95/98/NT. It is also the Fortran interface for Mesa.

OpenGL is a software interface for applications to generate interactive 2D and 3D computer graphics independent of operating system, window system, and hardware operations. GLUT is the OpenGL Utility Toolkit, a window system independent API toolkit for writing OpenGL programs. Mesa is a 3D graphics library with an API which is very similar to that of OpenGL.

The Fortran 90 bindings for OpenGL are an alternative to the older FORTRAN 77 bindings. By using the new features of Fortran 90, they define an interface to OpenGL that does not depend on any extensions to the Fortran standard and provide access to the full functionality of OpenGL. This provides the capability of robust, standard-conforming, portable user application codes, and increases the similarity between the Fortran and C interfaces to OpenGL. The Fortran 90 bindings were favorably reviewed by J3, the US Fortran Standards Committee, and officially adopted by the OpenGL Architecture Review Board in February 1998.

**Documentation** for f90gl including the official Fortran 90 bindings for OpenGL, installation instructions and user's guide.

**Software**, including the f90gl distribution and updates.

All the **latest information** on f90gl.

---

Just for fun! Here is an animation of the f90gl logo (123K mpeg) done using f90gl.

You can also see example images produced by the example programs in the f90gl distribution.

---

# Links to related sites

- OpenGL Home Page
- GLUT Home Page
- Mesa Home Page
- The Fortran Market
- The OpenGL Web Site with tons of links

---

f90gl was written by William F. Mitchell of the Mathematical and Computational Sciences Division at the National Institute of Standards and Technology with the advice and help of many people. f90gl was produced by an agency of the U.S. Government, and is not subject to copyright in the United States.

The mention of specific products, trademarks, or brand names is for purposes of identification only. Such mention is not to be interpreted in any way as an endorsement or certification of such products or brands by the National Institute of Standards and Technology. All trademarks mentioned herein belong to their respective owners. OpenGL is a registered trademark of Silicon Graphics, Inc.

---

Last change to this page: December 4, 2000 Date this page created: 1998 Contact: William Mitchell Home Page

# Win32 Fortran Compiler Comparisons

# Windows Programming

| | APF | CVF | FTN77 | FTN95 | G77 | INTEL | LF90 | LF95 | NAS | WAT |
|---|---|---|---|---|---|---|---|---|---|---|
| Bundled Tools | MRWE[1] | QuickWin | ClearWin+ | ClearWin+ | No | QuickWin | Wisk[2] | Wisk[2] | No | No |
| API Calls | Yes | Yes | Yes | Yes | No | Yes | Yes | Yes | Yes | Yes |
| API Header File | Yes | Yes | Yes | Yes | No | No | Yes | Yes[7] | No | Win16 |
| .... Prototypes | Yes[4] | Yes[4] | Yes[6] | Yes[6] | No | No | No[5] | Yes[7] | No | ? |
| .... Structures | Yes | Yes | No | No | No | No | No | Yes[7] | No | ? |
| .... Constants | Yes | Yes | Yes | Yes | No | No | Yes | Yes[7] | No | ? |
| Create DLLs | Yes | Yes | Yes | Yes | ? | Yes | Yes | Yes | Yes | Yes |
| Call 3rd Party DLLs | Yes | Yes | Yes | Yes | ? | Yes | Yes | Yes | Yes | Yes |
| COM support | No | Yes | No | No | No | No | No | No | No | No |

| Notes | |
|---|---|
| 1 | Pro Fortran also includes limited versions of the Superplot 2D and 3D graphing packages |
| 2 | Wisk is the Winteracter Starter Kit - a cut-down version of Winteracter |
| 4 | INTERFACE with extensions. It's often necessary to use intrinsics in the CALL statement to convert to C calling conventions. |
| 5 | Convert to C calling conventions using intrinsics in CALL statement. |
| 6 | Non-standard format. Allows automatic conversion of VAL and REF arguments. Also converts Fortran to C strings. Most calls use natural Fortran syntax. |
| 7 | Compiled module, but no source code. |

**(C) Polyhedron Software Ltd. (1998-2001)**

*Last Updated on 15/03/01*

*By John Appleyard*
*Email: john@polyhedron.com*

# Win32 Fortran Compiler Comparisons

# Execution Time Benchmarks

(C) Polyhedron Software Ltd. (1998-2001)

---

Download Benchmark source code (BENCH.ZIP 215K)

| | APF77 | APF90 | DVF5 | DVF6 | G77 | LF90 | FTN77 | FTN95 | PS4 | WAT |
|---|---|---|---|---|---|---|---|---|---|---|
| B1 | 2.62 | 2.60 | 2.84 | 2.80 | 2.91 | 2.58 | 3.00 | 2.12 | 2.56 | 2.52 |
| B2 | 5.82 | 8.69 | 6.70 | 5.23 | 8.86 | 5.80 | 6.90 | 5.56 | 5.94 | 6.30 |
| B3 | 1.22 | 1.15 | 0.97 | 0.96 | 1.14 | 1.14 | 1.37 | 1.05 | 1.08 | 0.99 |
| B4 | 1.19 | 1.21 | 0.95 | 0.96 | 1.05 | 1.14 | 1.33 | 0.97 | 1.01 | 0.91 |
| B5 | 1.48 | 1.40 | 1.37 | 1.34 | 1.95 | 1.36 | 1.79 | 1.47 | 1.17 | 1.29 |
| B6 | 3.25 | 3.27 | 4.25 | 2.72 | 6.63 | 3.49 | 3.64 | 3.46 | 2.99 | 2.76 |
| B7 | 37.93 | 33.15 | 30.14 | 25.60 | 43.51 | 31.18 | 39.53 | 33.79 | 28.31 | 29.95 |
| B8 | 38.12 | 32.70 | 29.22 | 25.13 | 43.03 | 29.92 | 39.25 | 33.05 | 27.75 | 28.95 |
| B9 | 1.76 | 2.15 | 1.69 | 1.65 | 2.41 | 2.01 | 3.26 | 2.98 | 2.45 | 1.90 |
| B10 (F90) | N/A | 28.43 | 25.66 | 24.41 | N/A | 36.76 | N/A | 33.96 | 27.00 | N/A |
| B11 (F90) | N/A | 32.12 | 23.42 | 20.21 | N/A | 29.43 | N/A | 49.22 | 23.58 | N/A |
| B12 | 6.52 | 11.08 | 9.55 | 9.25 | 7.83 | 8.23 | 9.51 | 6.75 | 7.24 | 7.96 |
| SM4 | 2.65 | 2.58 | 1.26 | 1.24 | 4.75 | 1.39 | 2.91 | 1.76 | 1.25 | 2.06 |
| SM8 | 2.69 | 2.62 | 1.25 | 1.24 | 5.13 | 1.40 | 2.97 | 1.82 | 1.30 | 2.09 |
| LM4 | 1.18 | 1.18 | 0.89 | 0.85 | 1.17 | 1.01 | 1.14 | 0.85 | 0.85 | 1.03 |
| LM8 | 1.36 | 1.41 | 1.13 | 1.11 | 1.43 | 1.27 | 1.80 | 1.14 | 1.09 | 1.28 |
| LP4 | 13.14 | 13.75 | 13.14 | 12.57 | 13.46 | 10.17 | 16.95 | 10.10 | 11.65 | 10.94 |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| LP8 | 9.52 | 9.70 | 9.20 | 9.40 | 9.87 | 7.96 | 12.90 | 8.59 | 8.44 | 8.38 |
| WH4 | 7.19 | 6.87 | 7.63 | 3.92 | 10.33 | 4.01 | 9.05 | 7.61 | 6.52 | 6.34 |
| WH8 | 6.90 | 7.12 | 7.61 | 4.07 | 11.41 | 4.25 | 9.03 | 8.30 | 6.46 | 6.33 |
| MDB | 49.07 | 50.00 | 45.17 | 42.69 | 51.71 | 37.78 | 60.93 | 56.18 | 42.60 | 46.12 |
| RO | 0.89 | 1.30 | 0.87 | 0.84 | 1.57 | 1.25 | 1.10 | 1.00 | 2.80 | 1.30 |
| AC | 43.55 | 49.88 | 74.67 | 70.60 | 77.98 | 46.23 | 52.68 | 41.13 | 71.78 | 39.20 |
| AIR | 167.42 | 228.35 | 150.14 | 149.74 | 244.96 | 147.91 | 203.59 | 208.56 | 159.68 | 159.18 |
| NB | 8.27 | 8.28 | 18.52 | 18.45 | 16.95 | 9.03 | 8.02 | 6.15 | 19.33 | 5.37 |
| PI | 7.44 | 7.92 | 20.70 | 19.73 | 15.09 | 7.32 | 5.30 | 4.11 | 14.70 | 4.70 |
| RAN | 3.83 | 3.67 | 3.44 | 3.44 | 3.57 | 3.09 | 5.08 | 3.85 | 3.78 | 3.62 |
| SSORT | 7.98 | 7.84 | 7.30 | 6.92 | 8.54 | 5.18 | 8.58 | 6.88 | 7.82 | 7.72 |
| TFFT | 11.12 | 12.06 | 11.31 | 10.94 | 11.45 | 10.72 | 12.99 | 11.46 | 10.47 | 10.46 |
| 2D | 6.68 | 9.10 | 11.92 | 11.66 | 11.83 | 6.74 | 6.53 | 5.50 | 10.43 | 5.29 |

| Compiler Switches | |
|---|---|
| APF77 | f77 %1 -O -N109 -N34 -stack:0x800000 |
| APF90[1] | f90 %1 -O -stack:0x01800000 |
| DVF | df %1 /fast |
| G77 | g77 %1 -O2 -malign-double |
| LF90[2,3] | lf90 %1 -o3 -tp -npca -nsav -nstchk -stack 8000000 |
| FTN77 | ftn77 %1 /opt |
| FTN95 | ftn95 %1 /opt |
| PS4 | fl32 %1 /Ox /G5 |
| WAT | wfl386 %1 /OX /FP5 |

| Notes | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|

| | |
|---|---|
| | All figures are Execution Times in Seconds - measured on a Pentium 200MMX with 64 Mbytes RAM, and running Windows 95. Each figure is the average over repeated runs lasting at least 5 minutes (much more for some). Measurement error is typically <1%. Green cells highlight figures within 10% of the fastest. Red cells indicate figures which are more than 150% of the fastest. |
| 1 | AIR and NB have to be compiled with -s (static allocation) |
| 2 | The performance of LF90 executables varies appreciably with compiler switches. The selected settings are as recommended by Lahey, and provide a good all-round balance. |
| 3 | A larger stack is specified for B10 and B11 |

---

*Last Updated on 24/01/01*
*By John Appleyard*
*Email: john@polyhedron.com*

# Intel/Linux Compiler Comparisons - Performed on Intel Pentium III Fortran 77 Execution Time Benchmarks

| | APF77 | G77 | LF95 | NAG | NAS |
|---|---|---|---|---|---|
| AC | 7.71 | 12.76 | 6.72 | 7.27 | 18.13 |
| ADI | 4.85 | 5.51 | 3.84 | 5.90 | 6.50 |
| AIR | 21.55 | 28.92 | 21.52 | 24.13 | 32.33 |
| CHESS | 23.69 | 19.58 | 21.88 | 20.27 | 38.29 |
| DODUC | 42.35 | 60.69 | 40.60 | 68.41 | 80.85 |
| LM8 | 12.49 | 12.37 | 13.09 | 12.90 | 33.42 |
| LP8 | 11.70 | 16.26 | 11.19 | 11.83 | 26.73 |
| MDB | 6.48 | 7.32 | 6.09 | 9.94 | 11.59 |
| MOLEN | 16.94 | 22.86 | 16.59 | 20.19 | 22.49 |
| PI | 14.11 | 38.78 | 20.90 | 7.38 | 10.20 |
| PNPOLY | 6.23 | 8.31 | 18.12 | 10.93 | 15.59 |
| RO | 10.56 | 20.91 | 10.80 | 19.31 | 26.76 |
| TFFT | 43.67 | 45.48 | 44.01 | 46.64 | 50.23 |
| WH4 | 21.15 | 27.78 | 18.56 | 23.50 | 39.82 |
| | | | | | |
| *Geometric Mean* | 14.001 | 18.807 | 14.809 | 16.046 | 24.181 |

| Compiler Switches | |
|---|---|
| APF77 | f77 -f -O -B100 %1 |
| G77 | g77 %1 -O2 -malign-double -funroll-loops -march=pentiumpro |
| LF95 | lf95 %1 --tpp --nchk --ntrace --nsav -O --nwarn -static |
| NAG | f95 %1 -O4 -dusty -save -ieee=full -Wc,-malign-double |
| NAS | f95 %1 -fast |

| Notes | |
|---|---|

All figures are Execution Times in Seconds - measured on a 850 MHz Intel Pentium III with 256 Mbytes RAM, and running Mandrake 7.1. Each figure is the average over at least 10 runs (much more for some). Measurement error is typically <1%. Green cells highlight figures within 10% of the fastest. Red cells indicate figures which are more than 150% of the fastest.

So far as possible, we have used the compiler switches which give the best overall results. We have not attempted to tune individual benchmarks, and, in particular cases, different switch settings may give better results.

These benchmarks are the same as those used to compare Win32 compilers on the same machine.

Download Benchmark source code (F77BENCH.ZIP 114K)

---

**(C) Polyhedron Software Ltd. (1998-2001)**

*Last Updated on 22/05/01*

*By John Appleyard*
*Email: john@polyhedron.com*

# Win32 Compiler Comparisons - Performed on an Intel Pentium III Fortran 90 Execution Time Benchmarks

| | APF95 | CVF | FTN95 | INTEL | LF90 | LF95 | NAS |
|---|---|---|---|---|---|---|---|
| CAPACITA | 397.74 | 369.52 | 351.71 | 359.30 | 397.39 | 381.32 | 488.01 |
| CHANNEL | 200.09 | 190.27 | 267.49 | 191.56 | 261.47 | 200.27 | 191.32 |
| FATIGUE | 101.27 | 64.86 | 133.88 | 146.20 | 299.44 | 66.00 | 118.67 |
| GASDYNAM | 82.56 | 65.36 | 93.92 | 96.89 | 88.24 | 75.82 | 71.76 |
| INDUCTANCE | 2.83 | 2.99 | 4.11 | 24.57 | 42.29 | 3.08 | 4.37 |
| KEPLER | 14.15 | 18.20 | 17.17 | 26.62 | 34.09 | 13.10 | 28.81 |
| MONTECARLO | 8.85 | 5.01 | 7.91 | 5.04 | 4.50 | 8.38 | 4.54 |
| PROTEIN | 17.90 | 13.22 | 20.79 | 15.12 | 17.73 | 16.80 | 25.54 |
| RNFLOW | 28.44 | 22.50 | 25.32 | 22.83 | 26.94 | 25.74 | 39.89 |
| SCATTERING | 66.95 | 22.64 | 33.04 | 72.62 | 28.29 | 22.91 | 92.72 |
| | | | | | | | |
| Geometric Mean | 38.950 | 29.773 | 40.366 | 49.047 | 54.720 | 32.413 | 46.055 |

| Compiler Switches | |
|---|---|
| APF95 | f95 %1 -O -Q100 -xINTEGER -stack:0x05000000 |
| CVF | df %1 /fast /opt:5 /link /stack:64000000 |
| FTN95 | ftn95 %1 /optimise /p6<br>*(slink was used to specify a stack size of 0x5000000)* |
| INTEL | ifl %1 /O3 /G6 /QaxK /link /stack:64000000 |
| LF90 | lf90 %1 -o3 -tpp -npca -nsav -nstchk -stack 24000000 -winconsole |
| LF95 | lf95 %1 -tpp -nchk -ntrace -nsav -nstchk -o1 -nw -stack 24000000 |
| NAS | f95 %1 -fast -heap 100000 |

**Notes**

All figures are Execution Times in Seconds - measured on a 850 MHz Intel Pentium III with 256 Mbytes RAM, and running Windows 2000. Each figure is the average over at least 10 runs (much more for some). Measurement error is typically <1%. Green cells highlight figures within 10% of the fastest. Red cells indicate figures which are more than 150% of the fastest. An earlier version of these benchmarks, run under Windows 98 (and with earlier compiler versions) is also available.

So far as possible, we have used the compiler switches which give the best overall results. We have not attempted to tune individual benchmarks, and, in particular cases, different switch settings may give better results. Compaq recommend /opt:4 as the norm, but suggest that /opt:5 be tried on a case-by-case basis, as it can help some programs but hurt others. Similar comments apply to other compilers.

Thanks are due to Jos Bergervoet for permission to use his CAPACITA benchmark,

and to Quetzal Associates for permission to use their CHANNEL, FATIGUE, GASDYNAM, INDUCTANCE, KEPLER, MONTECARLO, PROTEIN, RNFLOW and SCATTERING benchmarks. All the benchmarks have been modified slightly to fit into our benchmarking harness.

The benchmarks were also used to compare Linux compilers on the same machine.

Download Benchmark source code (F90BENCH.ZIP 689K)

---

**(C) Polyhedron Software Ltd. (1998-2001)**

*Last Updated on 04/04/01*

*By John Appleyard*
*Email: john@polyhedron.com*

# Win32 Compiler Comparisons

# Fortran 90 Execution Time Benchmarks

(C) Polyhedron Software Ltd. (1998-2001)

---

The current version of these benchmarks is available here.

Thanks are due to Jos Bergervoet for permission to use his CAPACITA benchmark, and to Quetzal Associates for permission to use their CHANNEL, FATIGUE, GASDYNAM, INDUCTANCE, KEPLER, MONTECARLO, PROTEIN, RNFLOW and SCATTERING benchmarks. All the benchmarks have been modified slightly to fit into our benchmarking harness. Thanks are also due to James Van Buskirk for supplying the DVF/Alpha results.

Download Benchmark source code (F90BENCH.ZIP 689K)

| | APF90 | DVF5 | DVF6 | EPC | FTN95 | LF90 | LF95 | DVF/Alpha |
|---|---|---|---|---|---|---|---|---|
| CAPACITA | 1324.78 | 698.80 | 636.56 | 1996.63 | 732.34 | 653.64 | 688.89 | 546.40 |
| CHANNEL | 528.01 | 205.56 | 164.85 | 249.61 | | 312.96 | 221.90 | 153.85 |
| FATIGUE | 768.43 | 183.07 | 154.22 | 381.52 | 294.07 | 645.76 | 160.82 | 103.38 |
| GASDYNAM | 140.40 | 125.73 | 123.30 | 132.72 | 156.95 | 137.83 | 121.61 | 59.91 |
| INDUCTANCE | 22.40 | 8.09 | 6.91 | 46.62 | 8.58 | 94.39 | 8.74 | 3.11 |
| KEPLER | 77.77 | 36.14 | 40.70 | 46.44 | 37.23 | 79.17 | 27.52 | 22.12 |
| MONTECARLO | 24.19 | 13.03 | 9.52 | 13.11 | 12.91 | 6.71 | 11.41 | 5.98 |
| PROTEIN | 57.67 | 63.17 | 59.80 | 113.52 | 42.03 | 69.22 | 70.81 | 21.57 |
| RNFLOW | 72.32 | 57.25 | 51.26 | 57.45 | 46.03 | 50.76 | 48.96 | 32.71 |
| SCATTERING | 198.99 | 102.24 | 99.85 | 171.86 | 64.82 | 61.77 | 40.03 | 47.25 |

| Compiler Switches | |
|---|---|
| APF90 | f90 %1 -O -Q100 -stack:0x800000 |
| DVF5 | df %1 /fast |
| DVF6 | df %1 /fast /opt:5 /arch:p6 /tune:p6 /link /stack 64000000 |
| EPC | epcf90 %1 /fast /arch=P6+ |

| | |
|---|---|
| ***FTN95*** | ftn95 %1 /opt /p6 |
| ***LF90*** | lf90 %1 -o3 -tpp -npca -nsav -nstchk -stack 24000000 -winconsole |
| ***LF95*** | lf95 %1 -tpp -nchk -ntrace -nsav -nstchk -o1 -nw -stack 24000000 |
| ***DVF/Alpha*** | df %1 /optimize:5 /tune:ev56 /arch:ev56 /fast /link /stack:64000000 |

| ***Notes*** |
|---|
| All figures are Execution Times in Seconds - measured on a 400 MHz Pentium II with 128 Mbytes RAM, and running Windows 98. Each figure is the average over at least 10 runs (much more for some). Measurement error is typically <1%. Green cells highlight figures within 10% of the fastest. Red cells indicate figures which are more than 150% of the fastest.<br><br>So far as possible, we have used the compiler switches which give the best overall results. We have not attempted to tune individual benchmarks, and, in particular cases, different switch settings may give better results. Digital recommend /opt:4 as the norm, but suggest that /opt:5 be tried on a case-by-case basis, as it can help some programs but hurt others. Similar comments apply to other compilers.<br><br>The DVF/Alpha results were obtained using Digital Visual Fortran 6.0B3 on a 533Mhz Aspen durango II, 21164A 164LX motherboard with 128 MBytes SDRAM and 2 MB L3 cache.<br><br>Other version numbers are: |

| ***Code*** | ***Product*** | ***Version*** |
|---|---|---|
| APF90 | Absoft Pro Fortran (f90) | 6.0 |
| DVF6 | DEC Visual Fortran | 6.0B |
| EPC | Edinburgh Portable Compilers EPCF90 | 2.0.4 |
| LF90 | Lahey LF90 | 4.50g |
| LF95 | Lahey/Fujitsu LF95 | 5.50b |
| FTN95 | Salford FTN95 | 1.53 |

*Last Updated on 24/01/01*
*By John Appleyard*
*Email: john@polyhedron.com*

# Intel/Linux Compiler Comparisons - Performed on an Intel Pentium III Fortran 90 Execution Time Benchmarks

| | APF95 | LF95 | NAG | NAS |
|---|---|---|---|---|
| CAPACITA | 397.07 | 374.82 | 425.11 | 486.18 |
| CHANNEL | 197.68 | 181.94 | 204.66 | 187.11 |
| FATIGUE | 100.07 | 69.37 | 103.03 | 123.99 |
| GASDYNAM | 81.30 | 73.81 | 94.18 | 71.68 |
| INDUCTANCE | 2.86 | 3.16 | 6.28 | 4.43 |
| KEPLER | 14.37 | 12.36 | 27.13 | 29.34 |
| MONTECARLO | 8.97 | 7.76 | 5.46 | 4.51 |
| PROTEIN | 17.61 | 19.52 | 16.60 | 26.40 |
| RNFLOW | 25.45 | 24.56 | 29.04 | 40.65 |
| SCATTERING | 67.08 | 24.64 | 32.84 | 105.32 |
| | | | | |
| *Geometric Mean* | 38.458 | 32.337 | 40.687 | 47.086 |

| Compiler Switches | |
|---|---|
| APF95 | f90 -O -B100 -xINTEGER %1 |
| LF95 | lf95 %1 --tpp --nchk --ntrace --nsav -O --nwarn -static |
| NAG | f95 %1 -O4 -Wc,-malign-double -ieee=full -unsharedf95 |
| NAS | f95 %1 -fast -heap 100000 |

| Notes | |
|---|---|

| | |
|---|---|
| | All figures are Execution Times in Seconds - measured on a 850 MHz Intel Pentium III with 256 Mbytes RAM, and running Mandrake 7.1. Each figure is the average over at least 10 runs (much more for some). Measurement error is typically <1%. Green cells highlight figures within 10% of the fastest. Red cells indicate figures which are more than 150% of the fastest.<br><br>So far as possible, we have used the compiler switches which give the best overall results. We have not attempted to tune individual benchmarks, and, in particular cases, different switch settings may give better results.<br><br>These benchmarks are the same as those used to compare Win32 compilers on the same machine.<br>Download Benchmark source code (F90BENCH.ZIP 689K) |

---

**(C) Polyhedron Software Ltd. (1998-2001)**

*Last Updated on 22/05/01*

*By John Appleyard*
*Email: john@polyhedron.com*

# Processor Comparisons on Win32 Compilers

***Table 1 - Processor Comparison.***

*Figures are geometric mean of benchmark timings taken from the F77 and F90 comparisons. Low numbers are best.*

| Compiler | Athlon 850 MHz | PIII 850 MHz |
|---|---|---|
| *F77 Compilers:* | | |
| **APF77** | 12.806 | 13.942 |
| **CVF 6.5** | 11.531 | 12.393 |
| **FTN77** | 20.529 | 20.670 |
| **FTN95** | 17.418 | 15.908 |
| **G77** | 17.878 | 18.039 |
| **INTEL** | 11.037 | 12.065 |
| **LF90** | 14.520 | 14.384 |
| **LF95** | 12.957 | 14.404 |
| **NAS** | 21.697 | 22.462 |
| **WAT** | 14.168 | 15.072 |
| *F90 Compilers:* | | |
| **APF95** | 34.400 | 38.950 |
| **CVF 6.5** | 25.647 | 29.773 |
| **FTN95** | 39.496 | 40.366 |
| **INTEL** | 49.895 | 49.047 |
| **LF90** | 52.642 | 54.720 |
| **LF95** | 29.359 | 32.413 |
| **NAS** | 42.683 | 46.055 |

***Table 2 - Benchmark Comparison.***

*Shows the variability of results (different benchmarks run better on different processors). Figures are the geometric mean of benchmark timings for all compilers tested. Low numbers are best.*

| Benchmark | Athlon 850 MHz | PIII 850 MHz |
|---|---|---|

| *F77 Benchmarks:* | | |
|---|---|---|
| ***AC*** | 8.283 | 7.944 |
| ***ADI*** | 5.709 | 5.094 |
| ***AIR*** | 25.444 | 24.630 |
| ***CHESS*** | 23.576 | 22.763 |
| ***DODUC*** | 36.891 | 46.865 |
| ***LM8*** | 22.475 | 14.729 |
| ***LP8*** | 7.719 | 13.043 |
| ***MDB*** | 6.684 | 7.729 |
| ***MOLEN*** | 16.439 | 19.613 |
| ***PI*** | 12.286 | 11.897 |
| ***PNPOLY*** | 10.698 | 9.744 |
| ***RO*** | 11.028 | 15.048 |
| ***TFFT*** | 54.101 | 45.685 |
| ***WH4*** | 19.302 | 20.951 |
| *F90 Benchmarks:* | | |
| ***CAPACITA*** | 445.628 | 390.046 |
| ***CHANNEL*** | 184.402 | 212.465 |
| ***FATIGUE*** | 102.871 | 117.078 |
| ***GASDYNAM*** | 62.904 | 81.349 |
| ***INDUCTANCE*** | 5.218 | 6.493 |
| ***KEPLER*** | 17.411 | 20.486 |
| ***MONTECARLO*** | 6.784 | 6.067 |
| ***PROTEIN*** | 17.913 | 17.799 |
| ***RNFLOW*** | 27.832 | 26.915 |
| ***SCATTERING*** | 38.557 | 41.681 |

| **Computer Specifications** | |
|---|---|
| ***Athlon 850MHz*** | 850 MHz AMD Athlon Thunderbird with 256 Mbytes RAM. |
| ***PIII 850MHz*** | 850 MHz Intel Pentium III with 256 Mbytes RAM. |

| ***Notes*** | |
|---|---|
| | All processors were running Windows 2000, with compiler switches set to optimise results for each processor. |

***(C) Polyhedron Software Ltd. (1998-2001)***

*Last Updated on 04/04/01*

*By John Appleyard*
*Email: john@polyhedron.com*

# Win32 Fortran Compiler Comparisons

# Debugger and IDE

| *IDE/Debugger Feature* | *APF* | *CVF* | *FTN77* | *FTN95* | *INTEL* | *LF90* | *LF95* | *NAS* |
|---|---|---|---|---|---|---|---|---|
| Fully Integrated Editor | No[1] | Yes | No[1] | No[1] | No | Yes | Yes | No |
| Fully Integrated Profiler | Yes[2] | No[3] | Yes | Yes | No | No | No | No |
| F90 Module Compatible Build[4] | Yes | Yes | No | No | No | Yes | Yes | No |
| Debug DLLs called from VB/Delphi | ? | Yes | No | No | No | Yes | Yes | No |
| Value on Hover | Yes[5] | Yes | Yes | Yes | No | Yes[6] | Yes[6] | Yes[7] |
| In Scope Variable Window | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| Display Array Sections | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| Display PARAMETER values | No | No | Yes | Yes | No | Yes | Yes | Yes |
| Break on Access | Yes | No | Yes | Yes | Yes | No | No | No |
| Break on Change | Yes | Yes | Yes | Yes | Yes | Yes | Yes | No |
| Conditional Break | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| Change Value | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| Evaluate | Yes | Yes | Yes | Yes | Yes | No | No | Yes |
| Call Stack | Yes | Yes | Yes | Yes | Yes | No | No | Yes |
| Step Over | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| Step Out of | Yes | Yes | No | No | Yes | No | No | No |
| Run to | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |

| *Notes* | |
|---|---|
| 1 | Editor and Debugger are separate applications |
| 2 | Subprogram level profiler |
| 3 | External Profiler can be started from IDE. |
| 4 | Build should compile minimal set of source files, in the right order, taking account of F90 module dependencies |
| 5 | Double-click on variable - then hover |
| 6 | Right-click on variable |
| 7 | Right-click on variable and select dropdown menu item |

**(C) Polyhedron Software Ltd. (1998-2001)**

*Last Updated on 15/03/01*

*By John Appleyard*
*Email: john@polyhedron.com*

# Win32 Fortran Compiler Comparisons

# Special Features

| | |
|---|---|
| APF | Pro Fortran includes 3 separate compilers, for F77, F90, C/C++ |
| | Built in AutoParallelizer |
| | Operates pre-processors from IDE |
| | Extended Precision (INTEGER*8) |
| | Vax / Unix subroutine compatibility library |
| | MAKEDEPEND - automatic dependency scanning |
| | Big Endian <-> Little Endian data conversion |
| | pre-compiled BLAS and LAPACK |
| | OpenGL header files |
| | HDF libraries |
| CVF | Tool for Automating creation of modules to access OLE/COM objects |
| | Array Visualizer |
| | Tool for creating Fortran COM servers |
| | Data conversion from/to big-endian IEEE, IBM, Cray, VAX |
| | OpenGL header files and examples |
| | cpp-style preprocessor |
| | Integration into Microsoft Developer Studio IDE |
| | FPS4-compatible language syntax and library routines |
| FTN95 | Can handle disk files larger than 4 Gig under Windows NT |
| | "CheckMate" Dynamic Run-time Checking Technology |
| | Extended Precision (REAL*10) |
| G77 | Compiler, run-time libraries, and documentation are available as freely redistributable source code |
| INTEL | Intel Math Kernel library |
| | Enhanced Debugger v4.5 |
| | cpp-style preprocessor |
| | Extended Precision (INTEGER*8 and REAL*16) |
| | FPS4-compatible language syntax and library routines |
| | Source substantially compatible with Compaq Visual Fortran |
| | Plugs into MicroSoft Visual Studio Development Environment |

| | |
|---|---|
| | Full support for Intel Internet Streaming SIMD Entensions |
| LF90/95 | Fujitsu Visual Analyser (Graphical Call Tree and Logic Flow Diagrammer + Static Analysis Tool) |
| | Fujitsu Scientific Subroutine Library |
| | AUTOMAKE |
| | Quad Precision (LF95) |
| | OpenGL header files, libraries, examples |
| NAS | Fortran Encyclopaedia F95 numeric libraries |
| | Precompiled F90GL modules |
| | ISO_Varying_string module (f90 secondary standard) |
| | ISO C Interoperability module (F2000 extension) |
| | ISO IEEE Exception Handling (F2000 extension) |
| | F2000 language extensions for ALLOCATABLE attribute |
| | 80-bit reals with full hardware support |
| | Automated Make |

*Note: The contents of this page is based on vendor claims only and has not been independently verified.*

---

**(C) Polyhedron Software Ltd. (1998-2001)**

*Last Updated on 15/03/01*

*By John Appleyard*
*Email: john@polyhedron.com*

# Win32 Fortran Compiler Comparisons

## Other Stuff

| ***Feature*** | ***APF***[17] | ***CVF*** | ***FTN77*** | ***FTN95*** | ***G77*** | ***INTEL*** | ***LF90*** | ***LF95*** | ***NAS*** | ***WAT*** |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | |
| Static Link Compatible with VC++ | Yes | Yes | No | No | No | Yes | Yes | Yes | Yes | No |
| Dynamic Link Compatible with VC++ | Yes | Yes | Yes | Yes | ? | Yes | Yes | Yes | Yes | Yes |
| Bundled C/C++ compiler | Yes | No | No | Yes | Yes | No | No | Yes | No | No |
| Compilation Time (seconds)[2] | | | | | | | | | | |
| …. As 1 file + INCLUDEs | 53.4 | 38.4 | 1.5 | 3.4 | 30.8 | 15.1 | 51.1 | 18.3 | 44.7 | 13.6 |
| …. As 143 files + INCLUDEs | 30.0 | 31.0 | 1.6 | 6.7 | 24.6 | 15.4 | 27.5 | 14.2 | 61.3 | 13.1 |
| Compiled Code Size | | | | | | | | | | |
| …. Object code size | 669K | 687K | 584K | 539K | 594K | 845K | 645K | 840K | 3533K | 545K |
| …. Executable File Size | 694K | 672K | 336K[4] | 362K[4] | 444K | 708K | 469K[3] | 1081K | 2416K | 437K |
| Huge Arrays - Lazy Allocation[5] | No | No[6] | Yes[7] | Yes[7] | No | No | No | No | No | No |
| | | | | | | | | | | |
| ***O/S Support*** | ***APF*** | ***CVF*** | ***FTN77*** | ***FTN95*** | ***G77*** | ***INTEL*** | ***LF90*** | ***LF95*** | ***NAS*** | ***WAT*** |
| | | | | | | | | | | |
| DOS extender | No | No | Yes[9] | Yes[9] | Yes | No | Yes[8] | No | No | Yes |
| Windows 3.1 | No | No | Yes | Yes | ? | No | Yes | No | No | Yes |
| Windows 3.1 + Win32s | Yes | No | Yes | Yes | Yes | No | Yes | Yes | No | Yes |
| Windows 95 / 98 | Yes | Yes | Yes | Yes | Yes | Yes[15] | Yes | Yes | Yes | Yes |
| Windows 2000 | Yes | Yes | Yes | Yes | Yes | Yes[16] | Yes | Yes | Yes | Yes |
| Windows NT / Intel | Yes[22] | Yes | Yes | Yes | Yes | Yes[16] | Yes | Yes | Yes | Yes |
| Windows NT / Alpha | No | Yes[10] | No | No | ? | No | No | No | No | No |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Linux | $$[11] | $$[19] | No | No | Yes | No | No | $$[14] | Yes | No |
| Mac | $$[11] | No | No | No | Yes | No | No | No | No | No |
| OS/2 | No | No | No | No | Yes | No | DOSX | No | $$[18] | Yes |
| Other | No | $$[12] | No | No | Yes | No | No | No | No | No |
| | | | | | | | | | | |
| ***Documentation*** | ***APF*** | ***CVF*** | ***FTN77*** | ***FTN95*** | ***G77*** | ***INTEL*** | ***LF90*** | ***LF95*** | ***NAS*** | ***WAT*** |
| | | | | | | | | | | |
| User Guide (Printed) | Yes | $$ | Yes | Yes | No | No | Yes | Yes | Yes | $$ |
| User Guide (On-Line) | Yes | Yes | No | No | Yes | Yes | Yes | Yes | Yes | Yes |
| Language Reference (Printed) | $$ | Yes | No | No | No | No | Yes | Yes | Yes[21] | $$ |
| Language Reference (On-Line) | Yes | Yes | No | No | No | Yes | Yes | Yes | No | Yes |
| Example Programs | | | | | | | | | | |
| .... Standard Fortran | Yes | Yes | Yes | Yes | No | No | Yes | Yes | No | Yes |
| .... Windows Programming | Yes | Yes | Yes | Yes | No | No | Yes | Yes | No | Yes |
| .... Interface to VB, Delphi, VC++ | VB | Yes | No | No | No | No | Yes | Yes | ? | No |
| | | | | | | | | | | |
| ***Support*** | ***APF*** | ***CVF*** | ***FTN77*** | ***FTN95*** | ***G77*** | ***INTEL*** | ***LF90*** | ***LF95*** | ***NAS*** | ***WAT*** |
| | | | | | | | | | | |
| Web updates | Yes | Yes | Yes | Yes | Yes | ? | Yes | Yes | No | Yes |
| Free email support | Yes | Yes | Yes | Yes | Yes[13] | Yes | Yes | Yes | Yes | No |
| Free telephone support | Yes | No | No | No | No | ? | Yes[20] | Yes[20] | No | No |

| ***Notes*** | |
|---|---|
| $$ | Product is available, but does not come with the Win32 package |
| 2 | Application has 16000 lines (+ INCLUDEs) in 142 subprograms. |
| 3 | Depends on data requirements. Executable file size could be reduced by using the -pack switch for DOSX executables (but not Win32) |

| | |
|---|---|
| 4 | An external DLL (approx 1.5M) is also required |
| 5 | Test (1GIG.FOR) involves access to a few hundred uniformly distributed elements in a 1 Gigabyte array. Compiler should allocate pages only for parts of array that are used, and should not thrash. |
| 6 | Test program sometimes works if it can create a 1 gigabyte swap file |
| 7 | Using "Virtual COMMON" facility |
| 8 | A single executable can be used for DOSX and Win32 command mode. The DOS Extender emulates most of the Win32 API. |
| 9 | DOS extender will not work under Windows NT. Separate compilers for DOSX and Win32. |
| 10 | A separate compiler for Alpha/NT is included in the professional version |
| 11 | Absoft also sell compilers for Macintosh and Linux |
| 12 | The same code base is used for Compaq's Unix and VMS compilers. |
| 13 | Not guaranteed! |
| 14 | Lahey also sell a Linux version of LF95 |
| 15 | Enhanced debugger is not supported on Windows 95 / 98 |
| 16 | Professional or Enterprise edition of MicroSoft Visual C++ 5.0 / 6.0 or Microsoft Visual Studio 5.0 / 6.0 must be installed for compiler to work |
| 17 | The 'Compilation Time' and 'Compiled Code Size' results are for APF95 |
| 18 | OS/2 version will be available in the near future |
| 19 | Linux Alpha only, free for personal and educational use |
| 20 | Only for Professional version |
| 21 | Standard Version comes with 'Fortran 90/95 explained' by Metcalf and Reid (while stocks last) |
| 22 | Windows NT 4.0 Service Pack 4 or later must be installed |

---

**(C) Polyhedron Software Ltd. (1998-2001)**

*Last Updated on 15/03/01*

*By John Appleyard*
*Email: john@polyhedron.com*